Universidade de Brasília
Faculdade de Tecnologia
Departamento de Engenharia Elétrica

Safe-Record: segurança e privacidade para registros eletrônicos em saúde na nuvem

Stefano M. P. C. Souza

Dissertação apresentada como requisito parcial para
conclusão do Mestrado em Engenharia Elétrica

Orientador
Prof. Dr. Ricardo Staciarini Puttini

Brasília
2016

UNIVERSIDADE DE BRASÍLIA
FACULDADE DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

SAFE-RECORD: SEGURANÇA E PRIVACIDADE PARA REGISTROS ELETRÔNICOS EM SAÚDE NA NUVEM

STEFANO MOZART PONTES CANÊDO DE SOUZA

DISSERTAÇÃO DE MESTRADO SUBMETIDA AO DEPARTAMENTO DE ENGENHARIA ELÉTRICA DA FACULDADE DE TECNOLOGIA DA UNIVERSIDADE DE BRASÍLIA, COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE.

APROVADA POR:

RICARDO STACIARINI PUTTINI, Dr., ENE/UNB
(ORIENTADOR)

RAFAEL TIMÓTEO DE SOUSA JÚNIOR, Dr., ENE/UNB
(EXAMINADOR INTERNO)

JULIANA PEREIRA DE SOUZA ZINADER, Dra., UFG
(EXAMINADORA EXTERNA)

Brasília, 07 de julho de 2016.

FICHA CATALOGRÁFICA

SOUZA, STEFANO MOZART PONTES CANEDO DE
Safe-Record: segurança e privacidade de registros eletrônicos em saúde na nuvem. [Distrito Federal] 2016.
xiii, 134p., 210x297 mm (ENE/FT/UnB, Mestre, Engenharia Elétrica, 2016).
Dissertação de Mestrado - Universidade de Brasília.
Faculdade de Tecnologia.
Departamento de Engenharia Elétrica.

1. Computação em nuvem	2. OpenEHR
3. Privacidade	4. Critografia homomórfica
I. ENE/FT/UnB	II. Título (série)

REFERÊNCIA BIBLIOGRÁFICA SOUZA, S. M. P. C. (2016). Safe-Record: segurança e privacidade de registros eletrônicos em saúde na nuvem. Dissertação de Mestrado em Engenharia Elétrica, Publicação PPGEE.DM - 636/2016, Departamento de Engenharia Elétrica, Universidade de Brasília, Brasília, DF, 121p.

CESSÃO DE DIREITOS

AUTOR: Stefano Mozart Pontes Canedo de Souza.

TÍTULO: Safe-Record: segurança e privacidade de registros eletrônicos em saúde na nuvem.

GRAU / ANO: Mestre / 2016

Stefano Mozart Pontes Canedo de Souza
Núcleo Rural do Torto, Trecho 01, Chácara 07
71.538-100 Brasília - DF - Brasil.

Dedicatória

Dedico este trabalho à memória de minha mãe, a professora Loide, que me ensinou o valor do conhecimento e da sabedoria. Àquela que, em vida, declarou-se honrada simplesmente em saber que me propusera a este desafio.

Agradecimentos

Agradeço a Deus, significado e sustentação de toda vida, pela bênção de sua doce e constante presença, por todo alento e pela esperança. Por permitir, em seu arranjo soberano, que eu tivesse esta oportunidade.

Agradeço à minha família, especialmente à amada esposa Talita, e aos filhos Tirza e Judá, por seu sacrifício e paciência, que me permitiram a dedicação necessária à conclusão deste projeto.

Agradeço aos colegas do STM, notadamente aos gestores Flávio Botelho, Celso Andrade, Danilo Bontempo e Ianne Carvalho, que, mesmo sob a pressão de administrar uma gritante escassez de pessoal, sempre se mostraram solícitos em fornecer os horários especiais e a licença que me permitiram dedicar-me ao PPGEE.

Preciso agradecer, ainda, aos professores do PPGEE, em particular ao professor Anderson Nascimento, por toda motivação e entusiasmo, sobretudo por me fazer acreditar que poderia ter sucesso no programa. E ao meu orientador, Ricardo Puttini, por todo apoio e direção na execução desse trabalho.

Resumo

O mercado de computação em nuvem tornou disponível a equipes hospitalares e acadêmicas um volume virtualmente ilimitado de recursos computacionais, os quais têm sido utilizados em atividades de pesquisa e desenvolvimento. Apesar das evidentes vantagens da computação em nuvem, o armazenamento e o processamento de registros eletrônicos em saúde (RES) nesses ambientes levantam preocupações genuínas com a privacidade e a segurança dos pacientes que, voluntária ou involuntariamente, têm seus dados expostos.

O compartilhamento do controle sobre os dados entre provedores e consumidores da nuvem é uma vulnerabilidade que não pode ser eliminada por uma solução técnica simples ou pontual. O risco de que o provedor de serviços faça mau uso de seu acesso privilegiado à infraestrutura da nuvem é conhecido como *insider threat*. Tal risco só pode ser eliminado se, mesmo tendo acesso a todos os dados, o provedor (desonesto, ou honesto mas curioso) não possa fazer qualquer uso deles. Isso pode ser alcançado se toda informação que trafegar, for armazenada ou processada na nuvem for encriptada com chaves indisponíveis naquele ambiente.

O Safe-Record, uma aplicação com arquitetura compatível com serviços do tipo *Platform-as-a-Service* (PaaS), foi construído para demonstrar a utilização de esquemas criptográficos homomórficos e com preservação de ordem. Os quais, permitem a execução de buscas e operações aritméticas simples sobre dados de um RES na nuvem, sem que sejam decifrados e sem que qualquer informação relevante seja revelada ao provedor. Os resultados sugerem que esse tipo de solução provê níveis adequados de segurança e privacidade, além de ser técnica e economicamente viável.

Palavras-chave: Computação em nuvem, OpenEHR, privacidade, criptografia homomórfica

Abstract

The cloud computing market makes a virtualy unlimited volume of computational resources available to medical and academic groups which are able to use such resources in research and development. In spite of the evident benefits of the cloud, the storage of Electronic Helath Records or the execution of machine learning algorithms over health data on the cloud raises genuine concerns over the privacy and even the security of patients who, voluntarily or not, have their data exposed.

Among the many vulnerabilities of a tipical cloud computing evironment, there is one that is specially dificult to handle with a simple tecnical solution: the shared control over the data between providers and consumers of the cloud. The 'insider threat' can only be satisfactorily dealt with if, even with full access to the data, a malicious or semi-honest provider would not be able to gather any information from it.

The Safe-Record, an application with an architecture compatible with tipical PaaS services, was built to demonstrate the use of homomorfic and order-preserving cryptographic schemes that allow the data to be processed in its encripted form on the cloud. This application manages Electronic Health Records modeled according to the OpenEHR standards, storing them under a combination of encryption schemes that allow simple queries and a few arithmetic operations to be run over the encrypted data on the cloud, without ever decrypting or revealing any relevant information. The results sugest that this solution renders adequate levels of security and privacy and is both technicaly and economicaly viable.

Keywords: Cloud computing, OpenEHR, privacy, homomorphic cryptography

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Siglas

AHE *Additively Homomorphic Encryption.*

DPE *Dense Probabilistic Encryption.*

IaaS *Infrastructure-as-a-Service.*

OPE *Order-preserving Encryption.*

PaaS *Platform-as-a-Service.*

RES Registro Eletrônico em Saúde.

RSAES-OAEP *RSA Encryption System - Optimal Asymmetric Encryption Padding.*

RSASSA-PSS *RSA Signature Scheme with Appendix - Probabilistic Signature Scheme.*

S-RES Sistema RES.

SRP *Secure Remote Protocol.*

1 Introdução

O desenvolvimento do mercado de computação em nuvem trouxe, ao alcance de agentes de médio e pequeno porte, o poder computacional antes acessível apenas a grandes órgãos governamentais e a grandes corporações multinacionais. Do ponto de vista do consumidor da nuvem, existe, pela primeira vez, a sensação de acesso simples, imediato e ilimitado a recursos computacionais [92].

Essa capacidade computacional virtualmente "ilimitada", evidentemente, pode ser usada em benefício particular, de uma comunidade local, de uma nação ou mesmo em benefício de toda comunidade internacional. É o caso, por exemplo, de pesquisas na área de saúde, que agora podem ser realizadas mais facilmente por equipes hospitalares, grupos acadêmicos ou agências governamentais – especialmente com o uso algoritmos de aprendizagem de máquina executados sobre dados clínicos ou genômicos gravados na nuvem [87, 20]. No entanto, a despeito do claro potencial positivo, persistem preocupações genuínas com a segurança e a privacidade dos pacientes que, voluntariamente ou não, têm seus dados expostos nesse processo.

Analisando a composição de um Registro Eletrônico em Saúde (RES), é fácil perceber que, caso exposto a agentes estranhos ao serviço de atenção à saúde, são postas em risco a segurança e a incolumidade física e moral dos pacientes [76]. Registros médicos revelam informações privadas sensíveis, provindas, por exemplo, de observação corporal (pressão arterial, peso etc.), do histórico médico (diagnóstico e posologia) e de dados levantados durante a anamnese (processo no qual o paciente relata ao médico o histórico de uma condição). Esses registros não apenas apresentam dados que podem ser usados contra a segurança física do paciente, tais como alergias e outras fragilidades, mas também revelam padrões de consumo, histórico de viagens, orientação sexual e outras informações de foro íntimo dos pacientes.

Mesmo em contexto diverso ao da saúde, ao decidir lançar mão de um serviço de computação em nuvem, o consumidor deve contrapor aos benefícios técnicos e econômicos os riscos a que sujeitará seu próprio negócio e os demais envolvidos (funcionários, clientes, fornecedores etc.) [59]. Ao serem questionados acerca de problemas que impedem ou atrasam a adoção da computação em nuvem pelas corporações, uma parcela significativa de gestores aponta como preocupações prioritárias os riscos de perda e vazamento de dados [3, 22].

Contra a perda de dados, exite uma série de técnicas baseadas em primitivas criptográficas, tais como *Proof of Data Possession* (PDP), *Proof of Ownership* (PoW) e *Proof of Retrievability* (PoR), que dão ao consumidor da nuvem ferramentas adequadas para averiguar a integridade e recuperabilidade de seus dados na nuvem [4]. Essas técnicas não apenas constituem um corpo literário maduro, mas também têm diversas implementações comerciais consolidadas [107, 124, 37].

Também existe vasta literatura versando sobre técnicas e artefatos projetados para minimizar os riscos de vazamento dos dados. A maior parte dos trabalhos, porém, está focada em detalhes de implementação e gerência da infraestrutura, e parte de um modelo de ataque externo, isto é, perpetrado por um agente externo contra os consumidores na nuvem [103]. Esses trabalhos geralmente apresentam soluções para ameaças ao modelo de serviços conhecido como *Infrastructure-as-a-Service* (IaaS), no qual o consumidor tem total controle sobre máquinas virtuais na nuvem [10]. Esse modelo de serviço exige do consumidor um elevado nível de *expertise* e esforço para gestão dos serviços. Em contrapartida, permite o emprego de uma vasta gama de técnicas de segurança [24, 74, 47].

São raros, no entanto, os trabalhos que apresentam soluções para um cenário onde o consumidor tem menor nível de controle sobre os ativos na nuvem - como é o caso em serviços do tipo *Platform-as-a-Service* (PaaS) [25]. Quando o provedor de serviços da nuvem é considerado como potencial atacante, restam poucas opções para o consumidor desse tipo de serviço, além de cifrar todos os seus dados antes de remetê-los ao servidor da nuvem [119, 70].O *2015 Information Security Breaches Survey* [6], estudo realizado pelo governo do Reino Unido como parte de sua estratégia nacional de segurança, apontou que, em um ano, 75% das grandes corporações naquele país tiveram incidentes de segurança da informação relacionados ao pessoal interno. A proporção é de 31% para pequenas organizações.

A despeito da solidez organizacional, os grandes provedores de serviços de nuvem estão sujeitos a essa vulnerabilidade, conhecida na indústria como *insider threat*, e têm grandes chances de ter funcionários realizando ou dando causa a acessos indevidos aos ativos hospedados em seus servidores. Do ponto de vista do consumidor da nuvem, um ataque perpetrado por um elemento interno ao provedor é um ataque do próprio provedor.

Nesse caso, existem dois cenários principais: 1) o provedor malicioso, que atua ativamente para acessar ou alterar os dados ou ainda interferir na operação dos sistemas do consumidor; e 2) o provedor semi-honesto (ou "honesto, mas curioso") que pode eventualmente usar seu acesso privilegiado para beneficiar-se de dados armazenados na nuvem. Um ataque do primeiro tipo é mais arriscado, pois é mais facilmente identificável – já que causaria ruptura no funcionamento normal das aplicações do consumidor. Além disso, uma vez descoberto, evidentemente, comprometeria seriamente o negócio do provedor. O

segundo tipo, no entanto, além de abstruso, explora uma das principais deficiências do mercado de computação em nuvem: a dificuldade de definir, de maneira clara e objetiva, os limites de responsabilização entre provedores e consumidores.

O controle sobre os dados certamente é compartilhado, pois o provedor sempre terá acesso privilegiado a qualquer elemento da infraestrutura da nuvem. Porém, são os consumidores (aqueles que estabelecem contratos de serviço com os usuários finais) que, em última instância, são responsabilizados civil e penalmente pelos danos causados em caso de vazamento ou mal uso dos dados. Essa assimetria entre nível de controle e responsabilização potencializa a *insider threat* e gera implicações técnicas e legais.

Diversos padrões e certificações de gestão da segurança da informação já lidam com a possibilidade de que um empregado lance mão dos ativos da organização de maneira indevida, especialmente a fim de realizar atividades maliciosas. Mas não é possível simplesmente estender boas práticas, protocolos ou até mesmo certificações de uma organização para outra. Do consumidor ao provedor, ou vice-e-versa.

Assim, consumidores depositários de informação sensível, tais como dados clínicos, de movimentação financeira ou informações relacionadas à segurança nacional, podem se ver impedidos de usufruir serviços de computação em nuvem. A menos que possam garantir, eles mesmos, os níveis requeridos de confidencialidade e privacidade na custódia dos dados. Isto é, os próprios consumidores precisam certificar-se de que, mesmo tendo acesso a tudo o que for transportado, persistido ou processado na nuvem, um provedor desonesto, ou semi-honesto, não poderá fazer qualquer uso dos dados.

1.1 Motivação

A motivação para o trabalho de pesquisa exposto nesta dissertação é a necessidade de uma solução que garanta segurança da informação e privacidade para usuários finais de aplicações na nuvem. A asserção avaliada é a de que, caso possa encriptar os dados sem prejuízo das operações típicas que precisa realizar sobre eles na nuvem, o consumidor pode manter o devido controle sobre a segurança de seus ativos e a privacidade de seus usuários, mesmo diante de um provedor semi-honesto.

As instituições prestadoras de serviços de atenção à saúde (IAS), por exemplo, operam sob regras bastante estritas de custódia de dados, pois são responsabilizadas pela segurança dos registros eletrônicos em saúde sob sua guarda e pela privacidade de seus pacientes. Essas regras geralmente se organizam em torno de conceitos como 'Proteção de Informação em Saúde' e 'Anonimização do Registro', introduzidos pelo *Health Insurance Portability and Accountability Act* (HIPAA) – norma estadunidense que tem sido usada como base para a legislação na área de informática em saúde em muitos países.

No Brasil, por exemplo, o HIPAA influenciou a elaboração do Troca de Informação em Saúde Suplementar (TISS), o padrão de trocas de informação em saúde implantado pela Agência Nacional de Saúde Suplementar [39].

De maneira bastante concisa, esses conceitos implicam que as IAS devem garantir que nenhuma informação em saúde será exposta a qualquer agente estranho ao atendimento ao paciente e, mesmo que seja, tal vazamento não pode ser utilizado para identificá-lo. Por isso, registros demográficos e outros elementos que levem à identificação do paciente devem ser separados dos registros clínicos. Note que esse tipo de restrição não se aplica apenas às estruturas de dados diretamente relacionadas a conceitos clínicos, mas também se estende àqueles dados administrativos, tais como os que informam o pagamento pelo serviço de saúde ou o consumo de material relacionado ao tratamento.

Esse tipo de legislação, por um lado, viabiliza o uso da forma eletrônica do registro em saúde, pois assegura aos envolvidos que, embora esteja em forma digital, o prontuário médico não estará menos seguro do que em sua forma física. Por outro lado, eleva os já pesados custos dos sistemas de atenção à saúde, ao adicionar alta carga de complexidade à administração dos registros. Essa discussão ganhou renovada importância após a promulgação do *Health Information Technology for Economic and Clinical Health* (HITEC), um ato do governo norte-americano que destinou cerca de vinte bilhões de dólares em incentivos para a implantação e modernização de sistemas RES.

Destaca-se, ainda, que, na medida em que a indústria de computação em nuvem avança em ganhos de escala, diferenciação tecnológica, penetração e dominância do mercado de computação corporativa, a manutenção de *data centers* nas IAS se torna cada vez mais impraticável. Daí a importância de apresentar respostas aos desafios enfrentados pelos agentes da indústria de atenção à saúde que, ao mesmo tempo em que se veem compelidos a consumir recursos computacionais nessa modalidade, também são restringidos pela legislação que regula a custódia de informações em saúde. As garantias acerca da confidencialidade da informação e da privacidade dos pacientes são a principal preocupação.

Como exposto anteriormente, já existe extensa literatura relacionada à segurança na nuvem. No entanto, a questão central na maior parte dos trabalhos na área é a proteção contra ataques de agentes externos. Faz-se necessário, portanto, ampliar essa discussão para cenários de ataques perpetrados pelo provedor da nuvem. Por outro lado, os trabalhos que lidam especificamente com a segurança de aplicações de RES tendem a propor uma forma encriptação dos registros focada no controle de acesso, ignorando ou simplesmente eliminando a possibilidade de se realizar qualquer computação sobre os dados na nuvem [27, 14, 50].

Qualquer solução que utilize a nuvem apenas como componente de persistência deve ser considerada com bastante cautela. Seria inviável, por exemplo, exigir que, para a

simples busca sobre os RES, o médico tenha que esperar pelo *download* e deciframento de todos os registros em sua máquina local. O poder computacional da nuvem deve ser utilizado pelo menos para a seleção dos registros de interesse e para extração de informação anônima (e.g. totalizações, médias e modas).

Em sua Estratégia do Registro Eletrônico de Saúde 2016-2019 [21], o Ministério da Saúde demonstra como um sistema RES é crucial na provisão de serviços de atenção à saúde e bem-estar dos indivíduos. A Estratégia está alinhada com o Plano Plurianual do Ministério da Saúde (PPA/MS 2016-2019), que tem entre os objetivos estratégicos "Implantar o e-Saúde no Brasil, com destaque para o Registro Eletrônico em Saúde (RES) e para os Centros de Inteligência para suporte às decisões dos gestores públicos e decisões clínicas dos profissionais de saúde".

O Registro Eletrônico de Saúde Nacional (RES Nacional) deve prover, gradativamente, interoperabilidade para diversas aplicações e serviços de e-Saúde. A Plataforma e-Saúde, por sua vez, consiste em um barramento de serviços SOA (*Service Oriented Architeture*) que permite a agregação da informação produzida nos pontos de atendimento do Sistema Único de Saúde (SUS), e a sua integração com os sistemas de abrangência nacional, especialmente o Cartão Nacional da Saúde e o RES Nacional.

A conceituação de registro eletrônico em saúde apresentada na Estratégia, inclui uma série de aspectos, entre eles:

- A qualidade do RES está ligada ao seu uso em larga escala, nos níveis local, regional, estadual e nacional, nos setores públicos e privados;
- A existência de modelos lógicos padronizados ou consensuais é condição para que os repositórios sejam integrados e que sistemas heterogêneos armazenem dados nos repositórios e os recuperem quando necessário;
- O RES tem que incorporar aspectos éticos e legais que estabeleçam os direitos de acesso a que tipo de informação e em que circunstâncias;
- O RES tem que incorporar mecanismos de segurança da informação e, portanto, de autenticação de indivíduos, profissionais e outros usuários.

Algumas funções essenciais de um RES apresentadas pelo Ministério são:

- Criar e manter os registros de saúde de cada indivíduo unificados;
- Capturar dados demográficos relevantes e a sua história;
- Permitir que o indivíduo ou paciente tenha acesso aos seus dados de saúde;
- Oferecer protocolos e evidências para apoio à tomada de decisão pelo profissional de saúde, na prescrição e no atendimento, incluindo alertas.

Esses conceitos e requisitos funcionais mínimos não se aplicam apenas ao RES Nacional, mas, como mostra o Manual de Certificação para Sistemas de Registro Eletrônico em Saúde v4.2 [32], da Sociedade Brasileira de Informática em Saúde, devem ser encontrados em qualquer sistemas RES.

Entre os benefícios esperados na adoção de um RES de abrangência nacional, está a criação de uma grande base de dados da qual se possa extrair informações importantes sobre prevalência de doenças, efetividade de tratamentos, adequação de protocolos, diretrizes e consensos, bem como os custos e benefícios associados. O documento de estratégia do Ministério mostra ainda que a informação de saúde coletada pelo RES forma um material poderoso para a análise e tomada de decisão para ações de promoção de Saúde, ao permitir entender o estado de saúde da população coberta, e dos fatores de risco de saúde associados à população analisada.

A implantação de um sistema dessa natureza, com tais benefícios associados, certamente contribui significativamente para a melhoria dos serviços de saúde no SUS. A execução desse plano estratégico, contudo, implica em altíssimos custos, uma vez que o Ministério tem de administrar uma enorme infraestrutura computacional. Para os próximos anos, o Ministério prevê investimentos de cerca de um bilhão de reais.

O documento de estratégia, no entanto, não indica qualquer possibilidade de que o Ministério faça uso de serviços de computação em nuvem. A adoção da nuvem, apesar de representar um potencial barateamento do sistema, sequer pôde ser considerada, tendo em vista os graves riscos associados à segurança da informação e à privacidade do usuário final.

Daí a relevância do tema e da proposta deste trabalho de pesquisa. O Ministério, bem como as instituições de atenção à saúde que utilizam sistemas RES, só podem considerar o uso de serviços de nuvem caso reste comprovada a segurança de um sistema dessa natureza na nuvem. Essa comprovação depende, entre outros fatores, da adequação dos mecanismos e técnicas propostos às restrições e aos requisitos próprios dos sistemas informação em saúde. O desenvolvimento de técnicas de segurança baseadas em mecanismos criptográficos aplicáveis aos modelos lógicos padronizados ou consensuais na área de informação em saúde é, portanto, bastante relevante.

1.2 Objetivos

O objetivo desta dissertação é apresentar a viabilidade de uma solução baseada em criptografia homomórfica para a segurança de registros eletrônicos em saúde na nuvem.

1.2.1 Objetivos específicos

Com o fim de alcançar o objetivo geral deste trabalho de pesquisa, foram estabelecidos, e perseguidos, os seguintes objetivos específicos:

- **OE-01** - Elencar falhas e vulnerabilidades comuns aos ambientes de computação em nuvem;
- **OE-02** - Examinar as propostas de segurança para sistemas RES na literatura recente;
- **OE-03** - Propor um método de encriptação dos dados que permita a execução do conjunto de operações essenciais a serem executadas na nuvem;
- **OE-04** - Construir uma aplicação que demonstre a solução proposta, em arquitetura compatível com serviços de PaaS oferecidos pelos maiores provedores;
- **OE-05** - Analisar, a partir desse experimento, a adequação da solução proposta aos requisitos mínimos de um sistema RES.

1.3 Metodologia

O trabalho de pesquisa que deu base a esta dissertação se dividiu nas seguintes fases:

1. Revisão de literatura em computação em nuvem;
2. Revisão de literatura em segurança de sistemas RES, com especial atenção aos sistemas modelados de acordo com os padrões OpenEHR;
3. Construção do modelo criptográfico;
4. Construção de um aplicativo para prova de conceito;
5. Validação da proposta e análise de resultados.

O método de investigação científica empregado neste trabalho é de natureza aplicada, com abordagens descritiva e qualitativa do problema selecionado. Levando em conta os objetivos propostos, este trabalho tem caráter exploratório, pois busca desenvolver uma nova abordagem para o estudo do problema e aplicar essa abordagem a um caso concreto com vistas à análise e aprimoramento de conceitos e ideias, bem como à proposição de problemas mais precisos para investigação posterior [38].

De acordo com Kaplan e Duchon [69], a natureza eminentemente aplicada da pesquisa no campo de sistemas de informação requer a combinação de métodos qualitativos e quantitativos. Eles asseveram que investigações quantitativas, geralmente realizadas por

meio de testes de hipóteses baseados em métodos estatísticos, são extremamente limitadas quando os resultados esperados ou as aplicações reais pretendidas são altamente dependentes do contexto. Eles recorrem ao trabalho de Yin [123], que versa sobre o método de pesquisa por estudo de casos, para mostrar que a pesquisa quantitativa deve ser precedida de uma investigação qualitativa, na qual o problema é melhor definido a partir da observação do contexto, dos hábitos e das necessidades dos envolvidos. Isso é ainda mais relevante em pesquisas exploratórias, nas quais novas hipóteses são levantadas. Essas hipóteses precisam de uma contextualização, por meio de investigação qualitativa, para a criação de modelos e hipóteses mais refinados, que podem, então, ser testados por meio de métodos quantitativos.

Fase 1 - Revisão de literatura em computação em nuvem

Para atingir o primeiro objetivo específico deste trabalho, a primeira fase da pesquisa teve caráter eminentemente descritivo. A investigação realizada visou apontar os problemas recorrentes na literatura de relevância na área de computação em nuvem. Esse trabalho inicial de pesquisa foi importante para delimitar o foco de investigação das fases seguintes, ao permitir a identificação, nos trabalhos examinados, de problemas ainda em aberto.

Esse tipo de análise descritiva tem como método primordial a observação, classificação e interpretação dos fatos [49]. A observação da realidade deu-se não apenas pela leitura e seleção criteriosa dos problemas descritos na literatura acadêmica, mas também na busca de referências e soluções apresentadas na comunicação corporativa dos agentes na indústria. Desse modo, foi possível confrontar o que tem sido apontado como barreira, do ponto de vista de segurança, à adoção do modelo de computação em nuvem com as soluções e produtos apresentados tanto na academia quanto na indústria.

O trabalho consistiu no exame minucioso dos artigos que trataram de segurança para computação em nuvem publicados nas seguintes conferências:

- IEEE SSP - Symposium on Security and Privacy;
- ACM CCS - Conference on Computer & Communications Security;
- USENIX HotCloud - Workshop on Hot Topics in Cloud Computing.

Tais conferências foram consideradas mais relevantes tanto por, historicamente, serem palco da publicação do "estado-da-arte" na área de segurança computacional, quanto pelo número considerável de citações aos trabalhos publicados nelas – considerando informações de ferramentas como CiteSeer (`http://citeseer.ist.psu.edu`) e Google Scholar (`http://scholar.google.com`).

Os artigos foram categorizados em uma taxonomia bastante simples, de acordo com o tópico central abordado:

- Computação remota ou distribuída;
- Armazenamento;
- Dispositivos, *user agents* e canais de comunicação;
- Proteção da identidade e privacidade.

Fase 2 - Revisão de literatura em segurança de sistemas RES

Essa etapa do trabalho se caraterizou por uma investigação de natureza qualitativa, cujo objetivo era aumentar a familiaridade com o problema específico e com as soluções encontradas na literatura. As ferramentas de investigação, mais uma vez, são a observação e interpretação dos fatos. A diferença é que, ao contrário da análise descritiva, o interesse não são os conceitos gerais ou abstratos, mas sim os casos concretos e as lições deles decorrentes [100].

Nessa fase, foram estudados trabalhos recentes que tratassem especificamente da mesma aplicação prática selecionada para experimentação: segurança de registros eletrônicos em saúde, includindo encriptação e execução de buscas e algoritmos de aprendizagem de máquina sobre os registros na nuvem.

O trabalho nesse estágio também envolveu a análise dos padrões de modelagem de informação em saúde do *framework* OpenEHR. A opção pelo OpenEHR se deu pela importância desse padrão no contexto específico do Brasil. O OpenEHR é o padrão de referência na Portaria 2.073/2011, do Ministério da Saúde, que regula o uso de padrões de interoperabilidade e informação em saúde no âmbito do SUS, em todos as esferas de governo, e para os sistemas privados do setor de saúde suplementar. Esse padrão também tem recebido considerável atenção na literatura de informática em saúde [9, 53, 73, 15].

Essa análise teve natureza mista: à abordagem qualitativa, empregada no estudo dos padrões, aliou-se uma análise quantitativa das entradas de dados definidas nos arquétipos curados pela comunidade OpenEHR [41], bem como dos arquétipos produzidos no âmbito do projeto do RES Nacional [21].

Os padrões OpenEHR se dividem em três grandes grupos, ou camadas. Na camada superior, o *Service Model*, encontram-se componentes relacionados à integração e orquestração de componentes construídos em arquitetura orientada a serviços. Na camada intermediária, estão os componentes de engenharia do conhecimento. Os padrões no *Archetype Model* definem os formalismos necessários para definição e representação semanticamente válida de informação clínica. Os padrões na última camada, o *Reference Model*, trazem as definições de tipos de dados mais básicos e de sua representação física nos diversos tipos de bancos de dados. Os padrões nessa camada foram o alvo da investigação qualitativa, que visava levantar os requisitos funcionais mínimos para a construção da prova de conceito.

A análise quantitativa se deu sobre os arquétipos e *templates*, os modelos de dados construídos com o ferramental do OpenEHR. Foram analisadas as entradas de dados que compõem os arquétipos publicados no repositório público do projeto OpenEHR [41], bem como os arquétipos desenvolvidos pelo Ministério da Saúde que integram o *template* operacional implementado na fase de experimentação deste trabalho. A intenção era levantar a proporção de dados textuais ou numéricos, e a correspondente proporção de tipos de operações (busca ou comparação textual, comparação numérica, soma, multiplicação etc) a serem realizadas na nuvem.

Com base na análise da natureza da representação física dos arquétipos, chegou-se à conclusão de que as operações de natureza numérica têm preponderância. Elas são necessárias para identificação de tipos de registros, de tipos de entradas de dados e para identificação do prontuário específico a que pertence cada registro. As operações numéricas também são necessárias para extração de informações anônimas, tais como totalização de registros de um certo tipo e médias de valores observados.

Fase 3 - Construção do modelo criptográfico

Todo o conhecimento angariado nas fases anteriores foi aplicado na elaboração da seguinte proposta para a segurança de sistemas RES na nuvem: usar primitivas criptográficas fortes para controle de acesso aos registros e usar esquemas criptográficos aditivamente homomórficos e esquemas com preservação de ordem para realizar as computações necessárias sobre esses dados cifrados na nuvem.

Entre os critérios utilizados na seleção de mecanismos para composição desse modelo, estão:

- O desafio de autenticação de usuário gravado na nuvem não pode ser usado para recuperação da senha ou roubo da identidade do usuário;
- A autenticação não pode se limitar a um desafio relacionado à senha, pois esta pode ser encontrada por meios estranhos ao sistema, especialmente quando o usuário usa a mesma senha em diversas aplicações;
- O protocolo de autenticação não pode ser utilizado como vetor de ataque contra o sistema;
- A autorização de acesso à informação clínica deve ser imposta por uma primitiva criptográfica: isto é, apenas os portadores da chaves criptográficas adequadas podem ler ou alterar o registro;
- A autorização não pode ser forjada nem refutada;

- O mecanismo de controle de acesso não pode inviabilizar o acesso emergencial ao prontuário do paciente;
- Todas as operações de cifragem e decifragem devem ocorrer na máquina do usuário;
- O sistema deve permitir a comparação numérica (buscas por intervalos) sobre as cifras na nuvem;
- O sistema deve permitir a operação de a soma sobre as cifras na nuvem;
- As operações de busca, *download*, decifragem e exibição do registro em tela não podem demandar um tempo de processamento e espera que degrade ou inviabilize a usabilidade do sistema.

Diante desses requisitos, um esquema de criptográfico simétrico, operando em modo autenticado, foi empregado para controle de acesso ao objeto de dados principal – o registro clínico. Esse sistema conferiu a velocidade necessária à leitura e gravação do registro de forma segura, com fortes garantias de confidencialidade, autenticidade e integridade. Além do desempenho, outro requisito específico considerado na escolha desse sistema foi a sua ampla adoção no mercado, de forma que seja possível utilizar bibliotecas criptográficas maduras, que tenham passado pelo devido escrutínio da comunidade de segurança. Esse último requisito também está associado à interoperabilidade da base de dados: se utilizasse uma implementação proprietária, interna, os dados não poderiam ser enviados para outro sistemas e decifrados em outro contexto.

Um esquema de criptografia com preservação de ordem foi selecionado para instrumentalização das operações de busca numérica. Essas cifras são utilizadas como índices para identificação e seleção de registros. Também foi selecionado um esquema aditivamente homomórfico que dá base a um protocolo de computação segura, usado para o cálculo de médias sobre os valores cifrados.

Fase 4 - Construção da prova de conceito

Os conhecimentos adquiridos também foram aplicados no projeto e construção do SafeRecord: a prova de conceito da solução proposta. A solução inclui uma aplicação para dispositivos móveis, utilizada no controle de acesso. Inclui uma extensão para navegadores de Internet, usada para validação do código entregue pelo provedor e para a execução de funções criptográficas na máquina do usuário final. Como elemento principal, conta com uma aplicação de RES que gerencia registros criados de acordo com o *template* operacional RAS-AB (Resumo de Atendimento em Saúde - Atenção Básica), criado no âmbito do projeto do RES Nacional. O arquivo de definição do *template* (RAS-AB.oet), encontra-se no Apêndice A.

Fase 5 - Análise do resultados

O sistema resultante foi, então, analisado, componente a componente, em termos de garantias de segurança, *performance* e características técnicas, especialmente as de maior impacto econômico. Foram analisadas as condições, ou requisitos mínimos, do ambiente para a manutenção das garantias de segurança do protótipo. Também foi considerada a facilidade de aplicação dessa proposta, tendo em vista a escolha, sempre que possível, de algoritmos e implementações padronizadas, disponíveis em diversas linguagens de programação e bibliotecas criptográficas.

Esse trabalho permitiu avaliar a viabilidade da solução proposta. Isto é, avaliar se a solução de segurança proposta é realmente aplicável a sistemas RES, respondendo aos requisitos de segurança da informação e privacidade dos pacientes sem impedir o atendimento aos requistos funcionais mínimos associados a tais sistemas.

1.3.1 Contribuições

Esta dissertação apresenta o Safe-Record, um sistema constituído por três artefatos de software independentes, construído para demonstrar o uso de criptografia homomórfica como alternativa viável para a segurança de aplicações de Registros Eletrônicos em Saúde na nuvem. Esse projeto se diferencia de outros aplicativos e protótipos apresentados na literatura [65, 85, 95] por, primeiramente, analisar a natureza física dos dados, de acordo com um padrão de modelagem adequado ao RES, para, a partir dessa análise, selecionar e aplicar os esquemas criptográficos mais adequados a esses dados e às operações que devem ser executadas sobre eles na nuvem.

A contribuição mais relevante, portanto, é uma análise detida dos padrões OpenEHR que permitiu a identificação dos tipos de esquemas criptográficos existentes mais adequados para sistemas RES modelados de acordo com tais padrões. Essa análise levou em conta um modelo adversarial específico, o do provedor semi-honesto, no contexto de aplicações implantadas em ambientes de nuvem do tipo PaaS, incluindo subtipos como mPaaS (*mobile PaaS*) e iPaaS (*integration PaaS*). A característica principal da análise nesse modelo adversarial é que, além das proteções comuns a todos os sistemas de informação, é necessário implementar medidas para dirimir o risco de que o próprio gestor do servidor onde roda a aplicação acesse e use indevidamente os dados.

O modelo proposto foi demonstrado com a aplicação de esquemas criptográficos relativamente novos e com poucos experimentos práticos na literatura, o que também representa uma contribuição. Os artefatos de software produzidos no decurso do trabalho, dessa forma, podem ser considerados como uma contribuição, uma vez que foram disponibilizados à comunidade na forma de software livre e aberto.

A extensão para navegadores, por exemplo, é um ótimo recurso para experimentos na área de segurança. Ela traz implementações de funções criptográficas complexas, tais como assinatura e certificação digital, criptografia simétrica em modos autenticados e sem autenticação, criptografia aditivamente homomórfica, criptografia com preservação de ordem, além de uma implementação da versão mais recente do protocolo SRP. O aplicativo para dispositivos móveis, utilizado para autenticação por dois fatores, tem uma interface de programação de aplicativos (API) bastante simples, e também pode ser utilizado por qualquer aplicação que implemente essa API. Esse aplicativo usa assinaturas digitais no padrão PKCS#1 (v2.2), que podem ser verificadas por qualquer biblioteca criptográfica moderna.

1.3.2 Limitações

O protótipo construído como objeto de experimentação manipula registros gerados de acordo com um *template* operacional OpenEHR elaborado no âmbito do projeto do RES Nacional, do Ministério da Saúde. Os tipos e estratégias de buscas realizadas sobre os dados são limitadas à realidade do *template* implementado.

Esta delimitação do escopo foi necessária para garantir uma análise prática da solução proposta. Por isso, o protótipo não tem conformidade total com o OpenEHR. Não foi construído um motor de interpretação de consultas em *Archetype Query Language* (AQL), linguagem definida pelo OpenEHR para buscas sobre os modelos clínicos [42]. Também não foram construídos todos os elementos de informação de suporte, tais como as funcionalidades de controle de versionamento, *log* de acessos, divisão por diretórios, produção de extratos e outros elementos definidos nos pacotes de mais alto nível do OpenEHR.

O protótipo também não tem conformidade total com as terminologias e ontologias utilizadas nos arquétipos referenciados pelo *template* implementado, tais como ISO 639-1 (que define a lista de idiomas), e a própria terminologia interna do OpenEHR, que descreve as entradas dos arquétipos. Essa limitação se deu por que não seria necessário ter a disposição todos os termos definidos nesses padrões para apresentar apenas um subconjunto reduzidíssimo deles.

Essas falhas de conformidade, entretanto, não têm impacto na demonstração da proposta em tela, pois as funcionalidades necessárias para que o sistema atenda a esses requisitos podem ser adicionadas sem que se faça qualquer alteração na maneira como os mecanismos criptográficos selecionados são empregados. A adição dessas funcionalidades não implica alteração no controle de acesso proposto. Também não requer alteração na maneira como os registros clínicos e demográficos são encriptados e armazenados, nem nas buscas e demais operações executadas sobre eles na nuvem.

Quanto aos modelos de entrega de serviço da nuvem, seria possível analisar diferentes condições e cenários de segurança, derivadas de diferentes grupos de funcionalidades e distintos níveis de controle sobre a operação dos sistemas na nuvem. Esta pesquisa não buscou explorar todas essas possibilidades, limitando-se a um estudo de caso específico, relacionado a serviços do tipo PaaS. Modelo em que o consumidor tem reduzidas opções de configuração e pouca influência sobre a operação do serviço.

O modelo adversarial considerado na análise da segurança da solução proposta é aquele em que o provedor pode, eventualmente, acessar os dados hospedados pelo consumidor em seus servidores. Os problemas distintivos desse cenário são a perda de confidencialidade da informação e os riscos à privacidade do usuário final. Por essa razão, a discussão apresentada no decorrer deste texto se detém com maior minúcia na relação entre os mecanismos propostos e as questões da confidencialidade da informação em saúde e da privacidade do paciente. Os requisitos de autenticidade e integridade, no entanto, também foram considerados em todas as fases da pesquisa e fizeram parte da seleção e análise dos mecanismos criptográficos, bem como da definição das respectivas estratégias de emprego.

1.4 Organização do trabalho

Como resultado do trabalho supracitado, esta dissertação traz, no Capítulo 2, uma revisão de literatura sobre a segurança na computação em nuvem. No capítulo seguinte, uma exposição sucinta do OpenEHR, o padrão de modelagem de dados para Registros Eletrônicos em Saúde (RES) no qual se baseia a solução proposta neste trabalho. O quarto capítulo apresenta os sistemas, protocolos e primitivas criptográficos selecionados para a solução, com destaque para os de Paillier [89] e Boldyreva *et al* [18]. O Capítulo 5 apresenta detalhes do projeto e construção do Safe-Record, seus componentes, e uma análise dos resultados obtidos com esta prova de conceito. O sexto capítulo, por fim, sumariza as conclusões da pesquisa e apresenta indicações para trabalhos futuros.

2 Segurança na computação em nuvem

Computação em nuvem é um modelo de negócios no qual um estoque de capacidade computacional é ofertado através de uma rede de alta capacidade, sob requisição e com provisionamento elástico, na forma de um serviço medido [92]. O maior apelo desse modelo é que os consumidores da nuvem podem transformar despesas de capital em custeio operacional, lançando sobre o provedor da nuvem todo o fardo da implantação e manutenção da infraestrutura que dá suporte à operação de seus sistemas. Evitam, consequentemente, os riscos e custos de sub ou superprovisionamento [93].

O conceito de computação como serviço não é recente, tendo sido discutido desde a década de 1960 [91, 71]. No entanto, apenas recentemente observou-se o contexto econômico adequado, no qual computação e comunicação integram o *core bussiness* da maior parte das organizações. A esse novo contexto econômico, aliou-se a maturação da tecnologia que dá suporte ao provisionamento remoto de recursos computacionais, permitindo a formação do mercado de computação em nuvem.

Entre os fatores habilitadores, podemos mencionar o desenvolvimento de virtualizadores, hipervisores e algoritmos estatísticos de multiplexação e distribuição de carga. Mas os fatores determinantes, certamente, foram a evolução da capacidade e o barateamento dos serviços de Internet, que permitiram que os *data centers* se instalassem longe dos grandes centros urbanos, barateando o custo das instalações e da energia elétrica. A combinação desses fatores permitiu o aumento sem precedentes na escala dos *data centers* que gerou o excedente de capacidade negociado no mercado de recursos computacionais que conhecemos como computação em nuvem [8].

Naturalmente, os diversos atores (veja a Figura 2.1) neste mercado têm interesses específicos, que resultam em visões diferentes quanto à segurança na nuvem [112]. Segurança, no sentido econômico lato, é o esforço para garantir a continuidade do negócio, e visa proteger seus valores (instalações, produtos, serviços, conhecimento, imagem corporativa etc.). Essa proteção se dá por meio de um processo de identificação, avaliação e administração de riscos de perda ou interrupção.

De acordo com Ross Anderson, professor de segurança computacional na Universidade de Cambridge, segurança computacional é definida como uma série de medidas tomadas

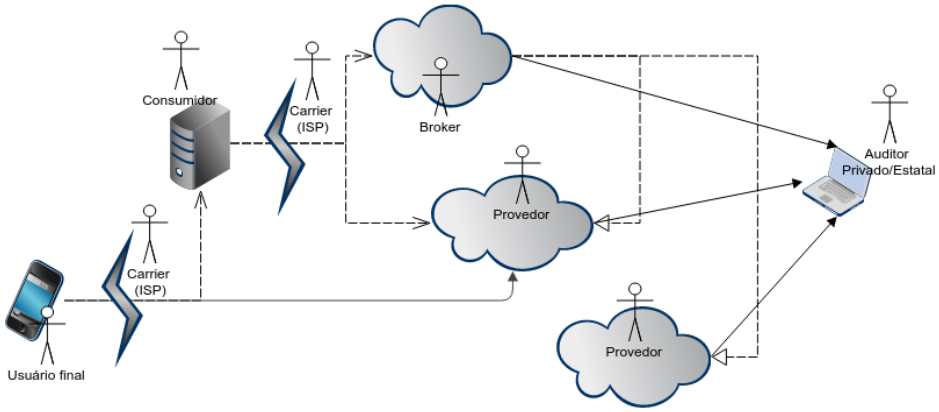

Figura 2.1: Principais atores do mercado de computação em nuvem

no curso de todo o ciclo de vida do aparato de software ou hardware (projeto, desenvolvimento/fabricação, implantação, manutenção e melhoria) com vistas a prevenir exceções à sua operação desejável. Ele assevera que quaisquer exceções, nesse contexto, podem ser definidas em termos de acesso ou manipulação não autorizados de dados, seja como resultado de malícia, erro ou eventualidade [5]. Isto é, a disponibilidade, integridade e confidencialidade da informação são os parâmetros básicos que definem a operação correta de sistemas computacionais.

A segurança na nuvem, num sentido amplo, pode ser vista como uma série de requisitos funcionais que precisam ser claramente definidos e implementados nas diversas camadas da infraestrutura de serviços com vistas à garantia das expectativas de negócio dos atores envolvidos. Esses requisitos influenciam, portanto, interfaces, APIs, configurações do ambiente, processos operacionais e gerenciais (tais como manutenção e auditoria), bem como os modelos de entrega serviços, entre outros aspectos da nuvem [64].

2.1 Definições básicas

Provedores de serviços de nuvem basicamente buscam garantir a capacidade de cobrar pelo uso de seus recursos. Para tanto precisam aferir adequadamente e ainda detectar e impedir anomalias de consumo. Evidentemente, precisam de um ambiente confiável, pelo qual os consumidores estejam dispostos a pagar. Provedores também estão sujeitos ao arcabouço legal do território em que operam, que pode exigir conformidade com uma série de padrões, certificações e procedimentos de auditoria. Segurança para provedores, portanto, está intimamente atrelada ao controle de acesso a seus ativos, bem como à manutenção de determinados padrões e níveis de serviço.

Consumidores, por sua vez, necessitam manter a governança sobre seus ativos, a despeito de terceirizarem o aparato tecnológico apoiando sua operação. Uma vez que a continuidade dos serviços prestados na nuvem está fora do controle dos consumidores, eles também precisam lidar com riscos de indisponibilidade por parte do provedor ou dos transportadores (ISPs que dão acesso à rede de alta capacidade ao provedor, ao consumidor ou aos usuários finais). Isso é importante pois, mesmo ao compartilharem o controle sobre a disponibilidade de seus sistemas com os provedores e transportadores, são os consumidores que, em última instância, são responsabilizados pela proteção de dados pessoais, privados e de identificação pessoal do usuário final. Segurança para consumidores da nuvem tem, portanto, um caráter mais atrelado à segurança da informação e à privacidade do usuário final.

Entidades governamentais, ao realizarem as funções típicas de Estado, buscam promover mercados estáveis, que contribuam para o desenvolvimento econômico e o bem-estar social. Por outro lado, também consomem um enorme volume de recursos computacionais, pois operam sistemas de alta criticidade, tais como aqueles usados para monitoração e regulação do sistema bancário, para administração fiscal e tributária, e para controle dos processos da justiça. Por isso mesmo, tais organizações são depositárias dos dados mais sensíveis de indivíduos e empresas numa nação, especialmente aqueles ligados à segurança nacional. Os entes governamentais, portanto, têm duplo interesse em promover o amadurecimento de tecnologias, agentes, bem como de padrões e boas práticas no mercado de computação em nuvem. Em benefício do próprio mercado, ou para que possam lançar mão de serviços confiáveis de computação em nuvem.

Outros atores, tais como auditores e corretores, têm sua participação no mercado praticamente determinada pela existência de padrões e procedimentos de segurança. Corretores ou *brokers*, como indicado na Figura 2.1, podem combinar serviços de diferentes provedores ou simplesmente agregar valor ao serviço de um único fornecedor. Em ambos os casos, garantias de nível de serviço, bem como os procedimentos de segurança, constituem uma parte essencial de seu negócio. Auditores, por sua vez, simplesmente não existiriam, neste contexto, não fosse por padrões e códigos de conduta aplicáveis aos serviços de computação em nuvem.

É importante notar também que, na maior parte das vezes, o usuário final dos insumos computacionais não é o consumidor da nuvem. O consumidor é a organização que paga ao provedor pelo uso dos recursos que, por sua vez, são consumidos por meio de aplicativos ou serviços disponibilizados ao usuário final. Este pode ser um membro da equipe da organização consumidora da nuvem, mas pode ser um agente externo a ela. Numa loja virtual, por exemplo, o consumidor seria a empresa mantenedora da loja, enquanto o usuário final seria o cliente que realiza operações comerciais com essa empresa. Em

situações como a desse exemplo, é muito comum que o usuário final não saiba que está se relacionando com mais de uma organização e que os dados confiados à loja, de certa forma, estão sendo entregues também ao provedor da nuvem. Segurança, do ponto de vista do usuário final, portanto, é garantir que somente aquelas organizações autorizadas tenham acesso a seus dados.

Essa discussão é útil para clarificar que segurança na computação em nuvem não pode ser vista de maneira reducionista. Por não se tratar de um paradigma computacional, nem de uma classe específica de sistemas computacionais, não pode ser reduzida a um modelo matemático universal, com o qual seja possível demonstrar provas de segurança e suas condicionantes. Por essa razão, os textos na área não apresentam soluções universais. Pelo contrário, em sua maioria, estão focados em ataques específicos, especialmente aqueles relacionados à invasão de máquina virtuais e à quebra de chaves criptográficas por meio de *side-channels* comuns em ambientes compartilhados.

Assim, para analisar a segurança de um ativo na nuvem, é necessário, primeiramente, escolher um ponto de vista: que interesses serão tutelados? Do provedor, do consumidor ou do usuário final? Ademais, a segurança de uma aplicação ou dos dados por ela manipulados não é determinada por uma mera agregação de mecanismos e tecnologias, mas sim por uma sólida compreensão das relações entre os ativos a serem protegidos e as ameaças tecnicamente viáveis e economicamente relevantes. Somente a partir dessa compreensão, pode-se produzir uma resposta adequada aos riscos reais [105].

> Unaligned incentives and information asymmetry better [statistically] explain systems flaws than technology issues.
>
> *Incentivos desalinhados e assimetria de informação melhor explicam [estatisticamente] as falhas de sistemas do que as questões tecnológicas.* (tradução livre) - Ross Anderson.

2.1.1 Arquitetura típica

Para entender os diferentes tipos de ativos que podem estar hospedados na nuvem e os diferentes ataques a que estão sujeitos, é necessário conhecer os modelos de serviço mais comuns. É necessário entender de que forma os diversos atores, inclusive usuários finais e agentes externos, podem interagir com os elementos constituintes de cada modelo de serviço. As formas de interação e os respectivos níveis de controle determinam o grau de participação dos diversos atores na garantia da segurança dos ativos na nuvem [112].

A Figura 2.2 mostra, de maneira concisa, o que seria a arquitetura típica de uma nuvem, apontando suas características essenciais, bem como as camadas de serviço mais comuns. No topo, os modelos de implantação: pública, comunitária, híbrida e privada. No quadro imediatamente inferior, as características de serviço que definem a computação

em nuvem: largo estoque de recursos computacionais; servido sob demanda; com rápida elasticidade; na forma de um serviço medido; através de uma rede de alta capacidade.

O modelo de implantação mais comum é o de nuvem pública, onde uma organização oferece capacidade computacional no mercado aberto. Uma implantação comunitária geralmente se refere à situação na qual um consórcio de organizações divide os custos de implantação e manutenção dos *data centers*, usufruindo dos recursos através de uma interface de provisionamento por serviço medido, típica da nuvem. Numa implantação privada, uma única organização arca com os custos do parque computacional, e suas unidades orgânicas consomem os recursos no formato de serviços de nuvem. E, finalmente, considera-se um modelo híbrido, quando o excedente de capacidade de uma implantação comunitária ou privada é oferecido ao público em geral.

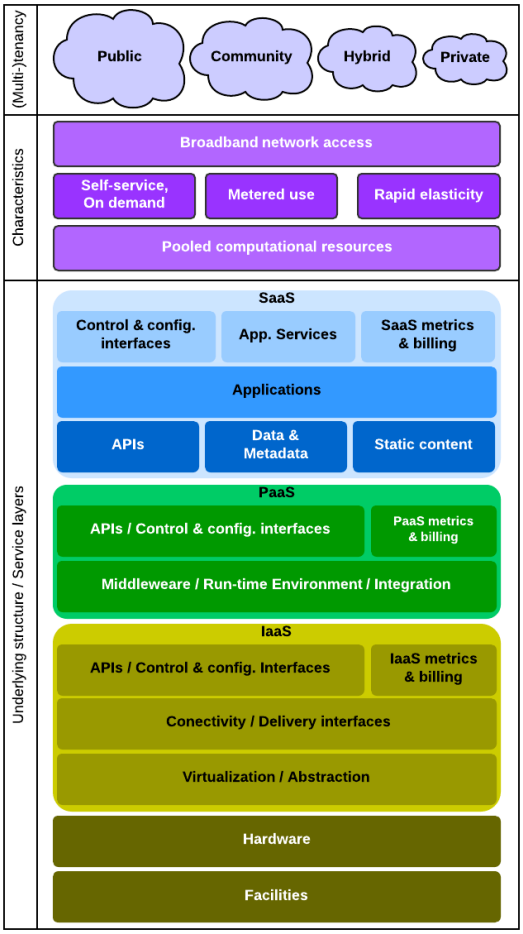

Figura 2.2: Características de uma nuvem

No terceiro quadro, vemos as camadas típicas de serviço. Os serviços ofertados na camada mais baixa são comumente classificados como IaaS e geralmente representam recursos mais básicos, representações diretas de aparatos físicos, tais como tempo de processamento, espaço em disco e tráfego na rede. Esses produtos são consumidos por meio de um conjunto de máquinas virtuais gerenciadas com o uso de uma interface específica criada pelo provedor. Essa interface deve oferecer funcionalidades tais como criação, destruição, ligamento, desligamento, cópia e configuração de máquinas virtuais.

Na segunda camada, encontram-se serviços mais complexos, tais como servidores de aplicação, sistemas gerenciadores de bancos de dados e outras ferramentas diretamente ligadas às necessidades de uma aplicação. Os serviços nessa camada são identificados como *Platform-as-a-Service* (PaaS) e atendem a consumidores com requisitos mais simples: precisam apenas de um ambiente para rodar uma aplicação, sem se preocupar com detalhes

do ambiente, tais como configurações do sistema operacional, de interfaces de rede, etc. O consumidor interage com o provedor através de uma interface de gerenciamento que, via de regra, disponibiliza funcionalidades tais como criar, disponibilizar, pausar ou remover uma aplicação, enviar e atualizar os fontes da aplicação e configurar limites de uso de recursos.

Na camada superior, por fim, estão os serviços de mais alto nível, classificados como *Software-as-a-Service* (SaaS). Nesse modelo de serviço, o consumidor interage diretamente com aplicações desenvolvidas pelo provedor do serviço, pagando pelo uso de funcionalidades específicas. Os mais expressivos (em volume de transações e valor comercial) são plataformas de vendas ou *Consumer Relationship Management* (CRM), especialmente as relacionadas a *e-commerce*, plataformas de *Enterprise Content Management* (ECM), de *Enterprise Resource Planning* (ERP) e *Enterprise Risk Management* (ERM). Companhias como Salesforce e SAP mantêm negócios bilionários nesse ramo.

Essa pilha de serviços na nuvem mostra que quanto maior a complexidade dos serviços, menor o nível de controle que resta ao consumidor. Ao abrir mão do controle sobre suas operações, entrega também o controle sobre a segurança de seus dados. Isso não quer dizer que estará mais exposto, pois, por outro lado, delega ao provedor, que provavelmente terá maior capacidade de investimento e pessoal mais capacitado, a responsabilidade de conhecer e responder adequadamente às diversas ameaças.

2.2 Principais ameaças e soluções

Existe uma relação direta entre o tipo ou nível do serviço de nuvem e o tipo de ameaças à segurança com as quais o consumidor deve lidar. Consumidores de IaaS devem se preocupar com vulnerabilidades típicas de ambientes compartilhados, ataques de negação de serviço, ataques às ferramentas de virtualização, e outras ameaças ao processamento dos dados. Consumidores de PaaS, por sua vez, estarão provavelmente mais focados na segurança dos dados, uma vez que cederam ao provedor o cuidado com a segurança dos processos. Consumidores de SaaS claramente têm um nível de confiança muito maior no provedor, preocupando-se apenas com a estabilidade dos níveis de serviços e das condições de negócio (disponibilidade, *performance*, custos, etc.) oferecidos pelo provedor [59].

2.2.1 Dispositivos e canais

Note que existe uma característica comum a todos eles: qualquer serviço de nuvem é gerenciado e consumido através da Internet. Na maior parte dos casos, a interface utilizada pelo usuário final é um aplicativo construído com tecnologia *web*, rodando num *browser* ou

em um dispositivo móvel. Por isso, todos têm de enfrentar ataques como *session injection*, *session fixation*, DNS hijacking, CSRF e XSS.

A providência mais básica é o fortalecimento dos procedimentos de autenticação e autorização com o uso de primitivas criptográficas, como, por exemplo, a implementação do protocolo SRP (Secure Remote Password), especificado na RFC 5054 [33]. Técnicas como identificação por dois fatores também podem elevar o nível de segurança das aplicações na nuvem [30].

Outras providências mínimas são o correto uso do protocolo HTTPS, combinado aos recursos de segurança padronizados presentes na máquina do usuário final, tais como *headers* HSTS (Strict-Transport-Security), CSP (Content-Security-Policy) e SRI (Subresource Integrity), que alteram a interação entre o servidor de aplicação e o agente de dispositivo do usuário, impedindo o *downgrade* das configurações de TLS e mitigando o risco de injeção de código malicioso no *browser*.

O potencial das vulnerabilidades de aplicativos *web* é ampliado quando os alvos atacados são justamente as interfaces de administração das contas do consumidor junto ao provedor da nuvem. Há diversos registros de ataques em que o controle do serviço é roubado explorando essas fragilidades. Outra falha de segurança ligada às interfaces de administração, especialmente para serviços do tipo IaaS e PaaS, é que elas são pouco intuitivas, sem uma padronização mínima em termos de conceitos e funcionalidades. Isso induz o consumidor a erros de configuração que acabam abrindo brechas para ataques contra sua conta de serviço ou contra sua aplicação [104].

Transporte

Aliam-se às fragilidades associadas aos agentes de usuário (e.g. *browsers*, leitores de e-mail e feeds RSS) aquelas associadas ao canal de transporte. O protocolo HTTPS, principal resposta às ameaças à comunicação entre provedores, consumidores e usuários finais, tem sua segurança drasticamente reduzida por descuidos comuns na implementação e no uso. Por um lado, a segurança do HTTPS depende da formação de um canal TCP seguro com o uso de outro protocolo – SSL ou TLS. E os servidores de aplicação e *browsers* dominantes no mercado dão mais importância à facilidade de uso do que aos requisitos de segurança e, por isso, durante os passos iniciais de negociação do protocolo, ainda permitem o *downgrade* para implementações antigas e provadamente inseguras do SSL.

Por outro lado, quando maiores cuidados são tomados na configuração do protocolo subjacente ao HTTPS, e são utilizados a versão mais recente do TLS, e uma combinação mais forte de criptossistema e função de *hash*, ainda restam fragilidades introduzidas pelo próprio protocolo HTTP ou pela forma como o HTTPS utiliza o canal TLS. O maior problema do HTTPS é sua dependência da '*web-of-trust*', o modelo de confiança cons-

truído em torno das infraestruturas de chaves públicas e certificados que dão significado às chaves RSA utilizadas para autenticar os *hosts* na *web* [28].

Browsers comerciais trazem uma lista pré-instalada de chaves públicas de autoridades certificadoras confiáveis, que são utilizadas para validar o certificado X.509 apresentado por um *web host*. É assim que o *browser* realiza a autenticação daquele servidor e a troca de chaves criptográficas que serão utilizadas para proteger as mensagens em uma sessão. Os ataques relacionados a esse sistema são vários: personificação da autoridade certificadora, falhas nas listas de revogação de certificados, falhas de implementação dos mecanismos de validação. Em muitos dispositivos móveis, o descuido é tão grave que apenas a validade do certificado é verificada, mas não sua correspondência ao *host* (domínio) que o apresentou [40].

Ataques contra os protocolos DNS e NTP, que raramente são configurados para trafegar por meio de um canal seguro, também podem levar um dispositivo a aceitar um certificado vencido ou aceitar um certificado válido emitido para um domínio diferente [78]. Além disso, existem vulnerabilidades relacionados a implementações específicas, que exigiriam apenas a atualização das bibliotecas criptográficas, mas que continuam tendo relevância pelo imobilismo dos consumidores e provedores. Um exemplo é a implementação dos protocolos TLS e DTLS (variante TLS para canais UDP) do OpenSSL, distribuído em milhões de *hosts* Linux, que mesmo após a publicação de ataques concretos (como Heartbleed e FREAK), ainda não foi atualizada em um número enorme de servidores [2].

Diante desse quadro é possível concluir que a segurança dos canais de comunicação depende de uma série de medidas que exigem total controle da configuração de servidores de aplicação, bibliotecas criptográficas, interfaces de rede e outros elementos dos sistemas operacionais: o que não é factível para a maioria dos consumidores de serviços IaaS, e simplesmente impensável no contexto de serviços PaaS e SaaS.

Consequentemente, é necessário considerar que qualquer informação em trânsito estará exposta, a menos que tenha sido anteriormente cifrada de maneira confiável no dispositivo do usuário e que as chaves utilizadas nessa cifra jamais transitem junto aos dados.

Identidade e privacidade

Novas tecnologias empregadas na construção de aplicações *web* de interface rica, como os padrões WebSQL, WebStorage, Web Sockets, Web Workers e Web Messaging do HTML5, e o novo formato de mensagens e *frames* binários do protocolo HTTP/2 também introduzem vulnerabilidades específicas. A possibilidade de gravar um grande volume de dados no cliente e abrir diversos *streams* binários a partir de uma única página HTML, ao mesmo tempo em que torna as aplicações bastante ágeis e responsivas, pode prejudicar a privacidade do usuários. Os muitos rastros deixados pela aplicação podem ser explorados

em técnicas avançadas de *device fingerprinting* para associar um dispositivo a um usuário de aplicação (fóruns, *webmails*, redes sociais, etc.) e, em última instância, revelar a identidade do portador daquele dispositivo [86].

Iniciativas como a extensão DNT (*do not track*) do protocolo HTTP e os padrões propostos no contexto do projeto PRISM (*PRIvacy-aware Secure Monitoring*), da União Europeia, tentam diminuir o impacto das ferramentas de análise de tráfego na privacidade dos usuários. A maior parte das iniciativas busca garantir a privacidade do usuário modificando uma ponta da comunicação, isto é, alterando a configuração e o funcionamento dos dispositivos do usuário. Isso pode ser visto em tecnologias amplamente difundidas, tais como os modos anônimos nos *browsers* e redes de *proxies* como a do projeto Thor. Todas essas iniciativas são, no entanto, insuficientes, pois o resultado depende também dos cuidados tomados por provedores de serviços e, principalmente, de uma atitude mais assertiva por parte do usuário [66].

O consumidor da nuvem geralmente tem baixo ou nenhum controle sobre a configuração ou o comportamento dos dispositivos dos usuários finais. Por isso, não dispõe de soluções para controlar a quantidade de informação que vaza para analisadores de tráfego na rede. Mas pode, no entanto, reduzir a superfície de exposição da identidade do usuário nas aplicações. Foi por essa razão que a justiça alemã determinou ao Facebook que desabilitasse funcionalidades como perfis públicos, busca por amigos e a listagem de pessoas que clicaram no botão 'curtir' associadas às publicações.

2.2.2 Processamento

As ameaças citadas até aqui estão relacionados ao caminho percorrido pela informação desde o dispositivo do usuário até o servidor na nuvem. Mesmo que a informação trafegue em segurança até a nuvem, durante o processamento estará exposta a diversas ameaças típicas de aplicações distribuídas e ambientes compartilhados. O tipo mais rudimentar é aquele relacionado ao descuido do programador que, muitas vezes, deixa de encriptar as mensagens trocadas entre o servidor de aplicação e o servidor de *storage* ou banco de dados – que, numa nuvem pública, pode estar até mesmo em outro continente.

Considerando que não haja esse tipo de descuido básico, ainda permanece o desafio de criar canais seguros de comunicação entre os diversos componentes das aplicações na nuvem. Entre as maiores dificuldades podemos citar o desafio de encontrar uma fonte de aleatoriedade forte e isolada para cada máquina virtual (VM, do inglês *virtual machine*) [72].

Essas fontes de aleatoriedade são um elemento fundamental de primitivas criptográficas e protocolos de processamento e comunicação seguros. A aplicação mais importante é justamente no contexto dos algoritmos geradores ou derivadores de chaves criptográficas.

Máquinas *desktop* contam com fontes como o estado elétrico de periféricos e o comportamentos do usuário, manifesto no número e endereço de processos ativos, posições de elementos gráficos ou do ponteiro do mouse no monitor. Um servidor, desprovido de periféricos e de interação direta com o usuário, não conta com essas fontes. Por isso, pode acabar gerando chaves com pouca entropia para cada VM ou chaves muito próximas, senão exatamente iguais, em diversas delas.

Outro tipo de vulnerabilidade muito comum diz respeito a falhas nos softwares de virtualização, monitoração e orquestração de recursos – ou na comunicação entre os elementos dessa pilha de software básico –, que podem ser exploradas para que o atacante obtenha dados de uma VM. A preocupação primária na academia tem sido a identificação e mitigação de ataques do tipo *side-channel*, que utilizam informação vazada por esses elementos para quebrar chaves criptográficas de VMs alvo.

Além disso, o provedor pode fazer uso malicioso de funcionalidades típicas do software que gerencia as VMs. Ele tem acesso, por exemplo, aos *snapshots* das VMs – visões completas do estado de uma máquina em certo instante, incluindo dados persistidos nos discos virtuais, o conteúdo da memória e buffers de rede. A ameaça de ataques internos, o *insider threat*, é potencializada pela ausência de instrumentos de auditoria e detecção de uso anômalo embarcados na própria pilha de software básico. Note que, mesmo que o ataque não seja perpetrado por um agente interno ao provedor, o simples vazamento de credenciais de um colaborador pode expor todos os consumidores [102].

Uma maneira de lidar com esse problema é simplesmente alterar a estrutura desse software, separando privilégios de administração geral (monitoração de estado de recursos físicos, habilitação de usuários e alocação de VMs para usuários, etc.) da administração das VMs (geração e encriptação de snapshots, copia de VMs, etc.) [24]. Outra solução é a inclusão de mecanismos mais fortes de auditoria, de forma que uma tentativa de abuso por parte de um agente administrador não possa ser escondida ou apagada [16].

Outra técnica, chamada de *VM nesting*, consiste em administrar máquinas virtuais rodando dentro das máquinas virtuais criadas pelo provedor. O consumidor se encarrega de gerenciar e coordenar a operação dessas máquinas e pode utilizar recursos como criptografia baseada em senha, de forma que, mesmo com total acesso ao primeiro nível, sem as senhas e chaves adequadas, um atacante não pode acessar o conteúdo do segundo nível [74]. Os servidores de aplicação e outros elementos dessa arquitetura de serviços mantida pelo consumidor também podem ser compilados e configurados de forma a melhorar a segurança, permitindo, inclusive, que o processamento se dê sobre uma versão cifrada dos dados [113].

Consumidores de serviços de *Platform-as-a-Service* não têm tamanho controle sobre o serviço, e, consequentemente, não podem utilizar técnicas tão complexas de defesa.

Mesmo assim, podem elevar o nível de segurança da informação diminuindo a superfície de exposição de suas aplicações. Uma técnica bastante comum é o particionamento virtual da aplicação, que consiste em criar barreiras – preferencialmente com uso de primitivas criptográficas fortes – de forma que a informação só tenha sentido no contexto de um componente específico da aplicação. Dessa forma, ainda que uma vulnerabilidade seja explorada em um ponto da aplicação, a informação produzida ou processada em outras partições permanece segura [47, 36].

Outra forma de particionamento lógico e encriptar todos os dados com chaves específicas para cada objeto de dado representado, ou para cada usuário. A aplicação de uma combinação de criptossistemas parcialmente homomórficos pode suprir todas as necessidades de processamento de aplicações mais simples. A informação cifrada é, então, armazenada e processada com uso de servidores de aplicação e sistemas de banco de dados padrão de mercado [35, 119].

Essa foi a estratégia adotada no decurso dessa pesquisa. A principal preocupação, ao se adotar esse tipo de procedimento, é selecionar os criptossistemas adequados ao domínio semântico da aplicação, isto é, adequados ao formato e natureza dos dados, bem como às operações essenciais a serem executadas sobre eles na nuvem. A intuição por trás dessa decisão é que o usuário nunca cede o controle dos dados, pois, ao cifrar os dados de maneira que seja possível realizar operações básicas, utiliza o poder computacional da nuvem sem que qualquer informação relevante seja desvendada ao provedor ou a qualquer atacante.

Existem trabalhos focados em aplicações que lidam com conteúdo textual, permitindo buscas por palavras-chave e até mesmo o estabelecimento de salas de *chat* [110, 96, 95]. Outros, focam em aplicações que lidam com dados estruturados, que exigem o uso de bancos de dados relacionais ou orientados a objetos [87].

2.2.3 Armazenamento

As maiores preocupações dos consumidores, e potenciais consumidores da nuvem, são a perda e o vazamento de dados. O vazamento de dados é, certamente, o problema mais complexo, pois depende da segurança de ponta-a-ponta: desde o dispositivo do usuário, passando pelos canais de transporte, pelo processamento na nuvem até o armazenamento. Não existe uma técnica ou produto que, sozinho, dê conta da segurança dos dados por todo esse percurso.

Contra a perda de dados, no entanto, existe uma série de técnicas que permitem ao consumidor uma avaliação precisa da qualidade e confiabilidade dos serviços do provedor. *Proof-of-Ownership, Proof-of-Possession* e *Proof-of-Retrievability* são técnicas que permitem que um consumidor com poder computacional limitado gere desafios não forjáveis

para que o provedor apresente provas do correto armazenamento dos dados [55, 17, 115]. *Oblivious Storage* é uma técnica que permite ao consumidor ocultar não somente o conteúdo persistido na nuvem, como também padrões de acesso e correlação entre os diversos blocos de dados [107]. É possível afirmar que esta é a área dos esforços de segurança na nuvem com resultados mais maduros, com diversos produtos comerciais já consolidados e amplamente disponíveis.

Outra ameaça, que talvez seja mais relevante mas que, no entanto, recebe menor atenção é quanto ao risco de *lock-in*. Isto é, a dificuldade de mover uma aplicação e seus dados para outro provedor de serviços. O consumidor sempre terá uma condição hipossuficiente na relação contratual com o provedor. Isto é, o consumidor é mero tomador de preços e jamais terá capacidade para garantir unilateralmente a manutenção os termos contratuais. Por isso, tem de lidar com o risco de que o provedor venha a alterar abruptamente o termos ou a qualidade do serviço. O provedor também pode simplesmente abandonar o mercado. Diante de todas essas situações, o consumidor pode ver-se desprovido de meios para garantir a posse e o acesso a seus dados.

A prática comum de tarifar o tráfego na rede mais pesadamente que o próprio espaço de armazenamento já induz ao *lock-in*. Além disso, sistemas de arquivos, bancos de dados, e até mesmo os formatos de máquinas e discos virtuais são distintos em cada provedor. Toda a pilha de serviços e tecnologias é bastante diferente entre provedores. Essas diferentes implementações visam facilitar e baratear a integração entre os diversos componentes da infraestrutura do provedor, mas funcionam também como instrumento de fidelização forçada do consumidor.

A Amazon, por exemplo, organiza serviços em "zonas" altamente especializadas, com pilhas de software otimizadas para processamento, armazenamento, ou entrega de dados (CDN). A Microsoft, segundo maior fornecedor de IaaS, organiza suas ofertas num formato mais próximo ao de SOs comuns, com todos os recursos em um mesmo pacote, numa mesma pilha de software básico. Cada um dos maiores provedores de PaaS, como Google e Heroku, mantém seu próprio formato de *container* para aplicações, APIs e bibliotecas próprias, e até mesmo um linguajar próprio para descrever e taxar os serviços.

Dessa forma, ao implantar uma aplicação na nuvem, o consumidor já deve ter em mente ferramentas que lhe garantam a capacidade de efetivamente escolher um novo provedor e trasportar seus dados a qualquer momento. Entre as técnicas presentes na literatura estão o uso de múltiplas nuvens (ou múltiplos serviços de nuvem) desde o projeto da aplicação [106]. Outra consideração importante é o uso de *frameworks* que permitam o desacoplamento entre a aplicação e o servidor de aplicação e demais características do ambiente [52]. Além disso, é possível projetar uma API genérica e construir adaptadores que traduzam essa API genérica para cada ambiente específico dos diferentes provedores

[57].

2.3 Considerações finais

O serviços de computação em nuvem representam não só um novo tipo de negócio, mas também uma nova e já importantíssima modalidade de organização e de distribuição de software. A maneira como nos relacionaremos com diversas aplicações, desde buscadores e redes sociais até controladores de *smart homes*, já é grandemente impactada pela evolução da computação em nuvem.

Entre as tecnologias mais importantes para a segurança da informação na nuvem, de acordo com o que revela a literatura na área, podemos citar:

- Recursos de autenticação ou gestão de identidade federados (tais como SAML, OAuth e OpenID);
- Métodos de autenticação por múltiplos fatores (integração de diferentes redes e dispositivos);
- Métodos de prova de recuperabilidade e posse de informação (PoW, PoR, etc.);
- Criptossistemas homomórficos;

Após uma extensa pesquisa bibliográfica, no entanto, permanecem questões em aberto, isto é, com poucos resultados publicados e com poucas ou nenhuma solução viável ou prontamente disponível no mercado. Não foram encontrados, por exemplo, especificações e análises formais de protocolos criptográficos projetados especificamente para ambiente de computação em nuvem. Protocolos que considerem, na construção de modelos adversariais e no desenho da interação entre os agente, as características dos diferentes atores nesse ambiente: provedores, consumidores, *brokers, carriers*, usuários finais e auditores.

Esse tipo de análise seria a base para o desenvolvimento de uma nova classe de bibliotecas criptográficas. Também seria a base para a construção de APIs e bibliotecas seguras e provadamente adequadas às novas arquiteturas de aplicações distribuídas que surgiram com a nuvem.

Um tópico especialmente relevante é quanto aos protocolos de estabelecimento/distribuição e administração de chaves criptográficas. Os protocolos atualmente mais difundidos tratam do problema de estabelecer uma comunicação segura entre dois ou mais agentes, sobre um canal inseguro. A topologia de uma aplicação rodando na nuvem introduz novos desafios, pois um mesmo "agente" na comunicação pode existir em múltiplas instâncias. É comum que diversas instâncias da mesma aplicação sejam iniciadas por meio da replicação de toda a pilha de memória. Em muitas ocasiões, diversas instâncias terão

de interagir com o mesmo usuário, pois recebem requisições através de um balanceador de carga. É necessário decidir, por exemplo, entre estratégias que envolvam a troca eficiente de chaves entre o usuário e as diversas instâncias (e das instâncias entre si), ou manter as chaves e realizar todas as operações criptográficas no dispositivo do usuário.

Com o fim de estudar esse problema mais profundamente, e propor um caminho viável para a segurança de aplicações com requisitos fortes de segurança e privacidade, foi selecionado um tipo específico de aplicação: o Registro Eletrônico em Saúde. Dessa escolha resultou o estudo da natureza dos dados nesse campo de aplicação e, consequentemente, das técnicas criptográficas mais adequadas. As informações levantadas e os teste realizados se encontram nos próximos capítulos deste trabalho.

3 O framework OpenEHR

O OpenEHR (Open Electronic Health Record) é um *framework* semântico para a modelagem de Registros Eletrônicos em Saúde (RES). Ele é formado por diversos padrões cujo foco é orientar a modelagem de conhecimento clínico de forma que as bases de dados tenham interoperabilidade funcional e semântica.

Nesse contexto, interoperabilidade funcional significa garantir que os registros possam trafegar entre diferentes sistemas e dispositivos. Interoperabilidade semântica é garantir que, uma vez compartilhado, um registro apresentará sempre a mesma informação, com a mesma estrutura e valor semântico, preferencialmente codificada dentro de uma ontologia padronizada, que respeite os requisitos semânticos do domínio representado [15].

3.1 Registros Eletrônicos em Saúde

De acordo com a Sociedade Brasileira de Informática em Saúde, um RES é um repositório de informação a respeito da saúde de indivíduos, numa forma processável eletronicamente. Um Sistema RES (S-RES), por sua vez, é o aparato utilizado para registro, recuperação e manipulação das informações de um Registro Eletrônico em Saúde [32].

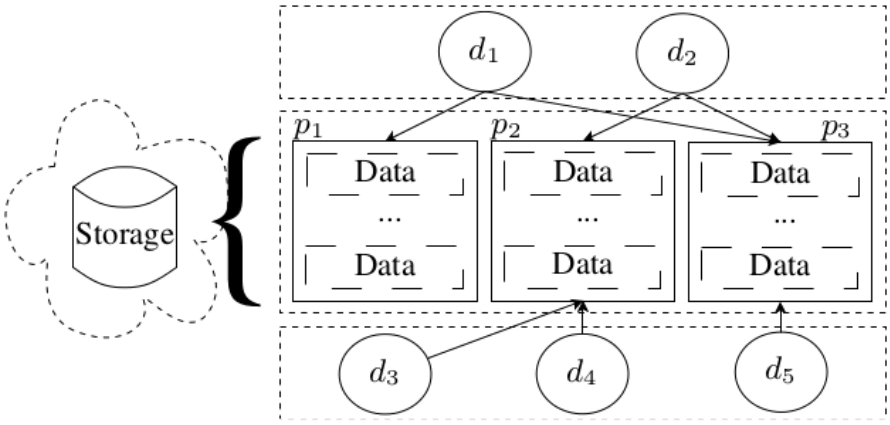

Figura 3.1: Representação de um RES [50]

Como mostra a Figura 3.1, o RES pode ser visto como um repositório onde se encontram os dados relativos à saúde de diversos pacientes ($p_1, p_2, p_3, ...$). Na maioria das

vezes, esse repositório é subdividido em prontuários individuais para cada paciente. E diversos médicos e outros profissionais da área da saúde ($d_1, d_2, d_3, ...$) acessam esses prontuários através de interfaces distintas e em tempos distintos, conforme a necessidade de atendimento ao paciente.

O uso de modelos lógicos consensuais para a representação de informação clínica é crucial para o atingimento desses objetivos de armazenamento e compartilhamento da informação. A qualidade dos modelos de dados influencia diretamente a qualidade do RES, isto é, a possibilidade de se extrair informação relevante do repositório [46].

O estabelecimento de um modelo consensual não apenas melhora a qualidade da informação registrada, mas também possibilita a integração entre diferentes sistemas [15]. Um caso de uso como o representado na Figura 3.1 (agregação de diversos prontuários, compartilhados entre diversos médicos) pode ser atingido por meio de um repositório centralizado, fixo, que é compartilhado por diferentes instituições e profissionais. Mas também pode ser atingido por meio de uma política de troca de informações, onde um sistema informa a outro sistema os dados necessários no contexto de um atendimento específico. O modelo consensual permite, ainda, que a instituição tenha a liberdade de escolher o melhor S-RES para sua realidade, a qualquer tempo, podendo inclusive utilizar sistemas abertos, desenvolvidos pela comunidade científica, e com boas garantias de qualidade [53].

3.2 Segurança de RES na nuvem

O propósito deste trabalho é propor uma solução de segurança, com foco no correto emprego de primitivas criptográficas, que atenda aos modelos consensuais na área de informática em saúde. Este exercício investigativo não se deu, evidentemente, sem o prévio exame de outras propostas já presentes na literatura. A maior parte dos trabalhos recentes que propõem o uso de primitivas criptográficas como base arquitetural do projeto e construção de sistemas RES seguros e com garantias de privacidade seguem a abordagem 'patient-centric': tomam o paciente como centro de todas as ações necessárias à criação e à guarda dos registros em saúde [109, 65].

Em [14], por exemplo, os autores propõem um esquema de criptografia hierárquica, de maneira que o próprio dado (uma estrutura arbórea) incorpore todas as regras de acesso. Estabelecem, ainda, que o paciente deve gerir o acesso aos diversos nós ou ramos (as diversas áreas) de seu prontuário, particionando-o em relação aos médicos a quem visita ou em relação às suas especialidades médicas: psiquiatria, cardiologia etc. Para tanto, o paciente deve derivar chaves específicas para cada nó, a partir da chave correspondente

ao nó imediatamente superior. A chave raiz, que dá acesso ao registro como um todo, permaneceria em posse do paciente durante todo o ciclo de vida do RES.

Esse tipo de estratégia centrada no paciente foi base para o projeto de vários produtos no mercado, alguns com relativo sucesso - como Indivo [80], o Google Health e o Microsoft Healthvault [108]. Um exame mais atento do domínio semântico dos dados aponta, no entanto, para outra direção: o prontuário, apesar de conceitualmente centrado no paciente, não deve ser controlado pelo paciente. A curatela do registro clínico por uma pessoa sem a devida formação profissional é inapropriada para a segurança da informação e do próprio paciente.

Primeiro, não é simples compartimentalizar ou fracionar informação em saúde. Os padrões de modelagem mantidos por profissionais da saúde (e.g. HL7 e OpenEHR) indicam uma estruturação do RES em torno de conceitos clínicos, o que exige a participação de um profissional habilitado tanto para modelagem quanto para gestão dos registros. Segundo, transferir a guarda do prontuário para o paciente, além de pouco natural – dado o histórico, comum no mundo inteiro, de guarda pela autoridade médica – pode até ser ilegal. É o que estipula, por exemplo, a legislação vigente no Brasil.

Ademais, são as instituições prestadoras de serviços de atenção à saúde (IAS) que são objeto da legislação e dos padrões de certificação relacionados a registros em saúde. Essas instituições são obrigadas a apresentar periodicamente às autoridades competentes diversas informações essenciais à regulação sanitária e do ato médico. Também são a fonte primária de dados que embasam a formulação de políticas públicas em saúde. Por isso, precisam ter acesso aos RES a todo tempo, a fim de produzirem as medidas estatísticas básicas requeridas para a produção de todos esses relatórios a que estão obrigadas.

Um RES estritamente *patient-centric* implicaria, portanto, ou na duplicação dos dados (uma cópia sob a guarda do paciente e outra com a instituição) ou em um processo complexo e demasiadamente custoso de deciframento, no dispositivo do paciente, e envio das informações requeridas pela IAS a cada ciclo de produção de relatórios. Em ambos os casos, o aumento significativo na probabilidade de vazamento dos dados acaba invalidando o esforço do paciente ao gerir o ciframento de seu prontuário.

Outro problema identificado na proposta de [14] é que o RES é encriptado de uma forma que anula completamente a possibilidade de se realizarem buscas e outras operações simples na nuvem. A preocupação com a segurança acabou suprimindo a preocupação com a garantia de requisitos mínimos de usabilidade. A maior parte das propostas sofre da mesma limitação, por, via de regra, encriptarem o RES como um bloco monolítico que será intratável na nuvem [50, 77].

Chase e Lauter apresentam um sistema bem mais amplo, que melhor se ajusta à realidade dos serviços em saúde. O projeto leva em conta diversos papéis desempenhados

na cadeia dessa indústria, desde operadoras de seguro, hospitais e clínicos autônomos, até farmácias e prestadores de serviços auxiliares [27]. O sistema garante a privacidade do paciente com o uso de um mecanismo de credenciais anônimas, na forma de *tokens* criptográficos. Esses *tokens* permitem, numa janela restrita de tempo, o acesso a informações específicas de um RES ou a troca de autorizações e endossos entre diferentes agentes, sem que a identidade do paciente ou qualquer outro dado (além do estritamente necessário no contexto de cada operação) seja revelado. Esse sistema é um modelo excelente de controle acesso e de arquitetura de autorização/endosso. Os autores não apresentam, no entanto, um modelo eficiente de manipulação dos registros na nuvem: o registros, mais uma vez, são encriptados como um bloco e nenhuma computação pode ser operada sobre eles na nuvem.

Em [20], Bos, Lauter e Naehrig apresentam métodos de uso de esquemas criptográficos homomórficos na execução de análises preditivas sobre RES encriptados, preservando, assim, a privacidade dos pacientes. Eles constroem um serviço que computa na nuvem as chances de um ataque cardíaco, a partir da análise de algumas observações corporais (altura, peso, pressão arterial). O paciente envia os dados referentes a uma observação corporal encriptados para o servidor, que responde com um resultado também cifrado. Apenas o dispositivo do paciente, que gerou a requisição, é capaz de decifrá-lo. Eles usam um modelo de classificação construído previamente, a partir de uma base de dados em claro.

O trabalho desses autores apresenta um excelente ramo de pesquisa, que, no entanto, difere da proposta deste trabalho de produzir uma solução que atenda o dia-a-dia clínico. A solução proposta nessa pesquisa pode funcionar como passo inicial, de captação dos dados numa forma controlada e segura e, futuramente, ser expandida de forma a prover meios para que, a partir da informação operacional básica, pesquisadores possam executar algoritmos de aprendizagem de máquina e construir modelos de classificação, regressão ou detecção de anomalias, na nuvem, sem jamais necessitar decifrar os dados dos pacientes.

O trabalho de Popa *et al* [95] utiliza o *framework* Mylar, construído pela equipe do segurança computacional do laboratório CSAIL, do MIT, como base para uma aplicação *web* que gerencia informações clínicas de pacientes de endometriose. O Mylar utiliza uma composição de dois sistemas de criptografia de chave pública. O primeiro dá base à Infraestrutura de Chaves Públicas (ICP) utilizada no controle de acesso, e é usado para criar uma cadeia de certificados e para encapsulamento das chaves utilizadas no segundo sistema. O segundo é o esquema MkSE (*Multi-key Searchable Encryption*) proposto em [96], e é usado na cifra do dados clínicos.

A segurança dessa aplicação depende da existência de um provedor de identidades confiável, não definido no trabalho. Apesar de utilizar uma ICP, o controle de acesso é

apenas lógico, isto é, embarcado no código da aplicação que controla o fluxo de requisições de um usuário autenticado. Dessa forma, uma vez autenticado, o usuário que pertencer ao grupo “Médicos", terá acesso a todas as chaves e, por conseguinte, a todos os dados dos paciente.

Essa arquitetura foi considerada insuficiente para uma aplicação real. Primeiro, por que o criptossistema utilizado na cifra dos dados permite apenas buscas de natureza textual. Mas, como será apresentado mais aditante, a representação física da informação clínica em padrões como OpenEHR e HL7, adotados por profissionais e autoridades da saúde, é majoritariamente numérica. Segundo, por não considerar questões como a privacidade do paciente (que exige algum isolamento entre os dados dos diversos pacientes) e a necessidade de garantias mais fortes de que apenas usuários autorizados terão acesso a um prontuário específico.

Tendo em vista essa insuficiência na área de segurança de S-RES na nuvem, isto é, a premente necessidade de propostas de segurança que fossem melhor ou mais facilmente integradas aos padrões consensuais de informática em saúde, o foco deste trabalho de pesquisa foi destinado ao exame de um padrão de modelagem de informação em saúde. Como discutido anteriormente, a opção pelo OpenEHR se deu pela importância desse padrão no contexto específico do Brasil, onde, por força de normativo do Ministério da Saúde, foi estabelecido como padrão de referência de interoperabilidade e informação em saúde. Além disso, o OpenEHR tem recebido considerável atenção na literatura de informática em saúde em âmbito internacional [73, 9, 53, 46, 15].

3.3 Padrões de informática em saúde

Todos os padrões do OpenEHR são livres e abertos, isto é, são produzidos e publicados como material *open source*, sob licenças como CC-BY-SA 3.0 (*Creative Commons Attribution-ShareAlike 3.0 Unported*). Entre os artefatos livres e abertos produzidos pela comunidade mantenedora do OpenEHR encontra-se ainda uma implementação de referência, que estabeleceu uma API padrão para a gestão dos registros médicos através de *templates* operacionais. A comunidade OpenEHR também tem colaborado com outras instituições de padronização. Um dos resultados desse tipo de cooperação é a série de padrões ISO/EN 13606, intitulada *Health informatics – Electronic health record communication* [61]:

- ISO 13606-1 - *Part 1: Reference Model*. Define um modelo de RES federado, no qual a informação clínica associada a um paciente (histórico de tratamentos, visitas médicas, observações corporais, instruções, etc) é compartilhada por um grupo de instituições ou profissionais;

- ISO 13606-2 - *Part 2: Archetype interchange specification.* Estabelece os formalismos utilizados para descrever (ou especificar) um arquétipo, bem como para realizar consultas sobre as estruturas de dados criadas sob esse modelo. Corresponde ao conteúdo dos padrões ADL e AQL do OpenEHR;
- ISO 13606-3 - *Part 3: Reference archetypes and term lists.* Traz um conjunto de arquétipos e terminologias que permitem a transmissão de um RES entre diferentes sistemas. As estruturas de dados definidas nesse padrão atendem tanto ao padrão OpenEHR quanto ao HL7 Version 3;
- ISO 13606-4 - *Part 4: Security.* Estabelece critérios ou políticas de acesso aos dados, de acordo papéis pré-estabelecidos;
- ISO 13606-5 - *Part 5: Interface specification.* Estabelece as interfaces básicas de comunicação utilizadas para transmitir um EHR_EXTRACT (um registro completo, com todos as instâncias de arquétipos referentes a um dado paciente), um RECORD_COMPONENT (a informação referente a um arquétipo ou entrada específicos) ou ainda um EHR_AUDIT_LOG_EXTRACT (registro de acessos, definido no ISO 13606-4);

Além dos padrões e da implementação de referência, o OpenEHR tem também um repositório de referência de conhecimento clínico, gerido através do *Clinical Knowledge Manager* (CKM), uma ferramenta de desenvolvimento colaborativo e governança dos artefatos daquele repositório [41]. Ali são mantidos arquétipos, *templates* e *templates operacionais* construídos e geridos de forma colaborativa pela comunidade. O repositório conta, atualmente, com mais de quatrocentos arquétipos, dos quais 47 publicados – fase de elaboração em que o arquétipo pode ser implantado em qualquer sistema em produção.

Existe uma variedade de padrões e organizações de padronização na área da saúde. A justificativa da escolha do OpenEHR para a realização deste trabalho de pesquisa se deu por sua importância como padrão de modelagem de dados, especialmente no Brasil. A Portaria nº 2.073/2011, do Ministério da Saúde, regulamentou o uso de padrões de interoperabilidade no Brasil, na esfera do SUS, em todos as esferas de Governo e no Sistema de Saúde Suplementar. Os padrões adotados nessa Portaria foram o OpenEHR, como a referência principal, a ser suplementada por padrões como HL7-CDA, SNOMED CT, DICOM e LOINC.

A Tabela 3.1 dá uma ideia do propósito de cada um desses padrões. É possível notar que cada um é especializado em um tema ou necessidade específica: enquanto o ISO 13606-2 cuida da descrição, à priori, dos modelos de dados, o HL7-CDA cuida da exportação de dados, independente do modelo de origem, cumprindo o papel de formato intermediário entre repositórios. O SNOMED CT é uma terminologia clínica geral, enquanto LOINC,

DICOM, ISBT 128 e ICD-10 são terminologias especializadas. O IHE-PIX, por fim, é um sistema de referenciamento que permite identificar e cruzar as informações de um paciente em diversos sistemas.

A Tabela 3.2 apresenta normas relevantes para a área de informática em saúde no Brasil. O conhecimento, ainda que superficial, da legislação e dos padrões correlatos foi extremamente relevante na formação da convicção necessária acerca da importância e do papel do OpenEHR nesse contexto.

Padrão	Descrição
ISO 13606-2	Padrão de interoperabilidade de modelos de conhecimento, incluindo arquétipos, *templates* e metodologia de gestão. Representa, na verdade, um subconjunto do pacote AM do OpenEHR.
ISO-HL7 CDA	HL7 *Clinical Document Architecture*, define uma estrutura semântica mínima utilizada como formato de referência para exportação de dados em boa parte dos sistemas RES (S-RES). Esse padrão é extremamente importante por constituir uma "ponte" de interoperabilidade entre os padrões na área que, na sua maioria, definem alguma forma de exportação dos dados para o HL7-CDA.
SNOMED CT	Uma terminologia padronizada, um *thesaurus* de termos clínicos, mantido pela International Health Terminology Standards Development Organisation. O padrão inclui mais de 311 mil termos, os códigos correspondentes, suas definições, além de uma taxonomia que inclui relacionamentos (sinônimos, especializações, generalizações, etc.) e classificações temáticas.
LOINC	*Logical Observation Identifiers Names and Codes*, é uma terminologia padronizada, nos moldes do SNOMED CT, mantida pelo Regenstrief Institute como um padrão *open source*. Diferentemente do SNOMED, no entanto, é especializado na temática laboratorial: testes, medidas, observações, etc.
DICOM	*Digital Imaging and Communications in Medicine*, é um padrão para representação e transmissão de informação relativa a exames de imagem.
ISBT 128	Padrão de codificação de dados de identificação das etiquetas de produtos relativos ao sangue humano, de células, tecidos e produtos de órgãos.
ICD-10	*International Classification of Diseases*, é uma terminologia mantida pela Organização Mundial da Saúde, especializada em doenças.
IHE-PIX	Especificação para o cruzamento de identificadores de pacientes de diferentes S-RES.

Tabela 3.1: Padrões relevantes de informática em saúde

Norma	Descrição
Portaria MS-2.072/2011	Estabelece o Comitê Gestor de Informação e Informática em Saúde, responsável por promover a evolução e a padronização de S-RES no Brasil.
Portaria MS-2.073/2011	Regulamenta o uso de padrões de interoperabilidade e informação em saúde no âmbito do SUS, bem como para o setor privado, no sistema de saúde complementar.
ANS TISS	Troca de Informação em Serviços de Saúde é o padrão estabelecido pela ANS para comunicação entre prestadores de serviços (hospitais, planos de saúde, etc) e destes com a Agência.
Res. CFM-1638/2002	Define Prontuário Médico e estabelece o conjunto mínimo de informações, bem como a responsabilidade de criação, manuseio e guarda. Estabelece ainda as atribuições da Comissão Revisora.
Res. CFM-1821/2007	Estabelece as regras para a digitalização do prontuário e determina sua guarda permanente, bem como a guarda de uma cópia impressa por 20 anos.
ISO/DIS 27.799	Orienta o correto emprego dos controles de segurança descritos na norma ISO/IEC 27.002 para um S-RES.
ABNT ISO/TR 20.514	Informática em saúde - Registro eletrônico de saúde - Definição, escopo e contexto.
ABNT ISO/TS 18.308	Informática em saúde - Requisitos para uma arquitetura do registro eletrônico.

Tabela 3.2: Legislação vigente no Brasil

O OpenEHR surge como principal padrão na legislação por que é o único que apresenta uma proposta generalista, que olha para formulação e gestão do RES como um todo, em todo o seu ciclo de vida. Enquanto a maior parte dos padrões trata apenas da terminologia, ou do formato de exportação dos dados, o OpenEHR se propõem a modelar desde a

linguagem que expressa a estrutura básica dos dados, passando pela linguagem de pesquisa sobre as bases de dados, até a API de um sistema distribuído numa arquitetura orientada a serviços.

Ademais, essa abordagem com foco no domínio semântico, apesar de relativamente nova, representa uma tendência irreversível, pois resulta da experiência e trabalho de uma ampla gama de profissionais da área da saúde e da indústria de informática em saúde. Como prova, observa-se que o OpenEHR, a despeito de sua complexidade e do caráter incipiente de vários de seus componentes, já é o padrão adotado por órgãos centrais dos sistemas de saúde em diversos países, tais como Austrália, Reino Unido, Rússia e Brasil [73].

3.4 Os componentes do OpenEHR

A Figura 3.2 mostra os três grandes grupos de padrões e demais artefatos que compõem o OpenEHR. No grupo de mais alto nível, o *Service Model* (SM), estão os componentes relacionados à arquitetura orientada a serviços. O esforço conjunto em torno da definição de uma camada de serviços padronizada decorre da percepção da comunidade OpenEHR de que o padrão arquitetural SOA (*Service-Oriented Architecture*) é a abordagem mais adequada para a construção e integração de sistemas de informação que respondam à complexidade da gestão de conhecimento em saúde [11].

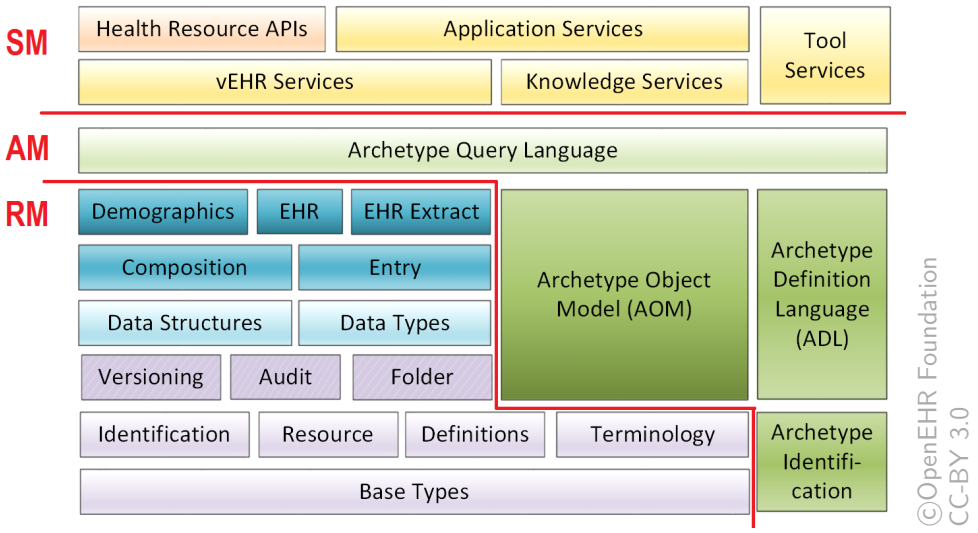

Figura 3.2: Pacotes de modelos e padrões OpenEHR [43]

Essa orientação se justifica pela necessidade de uma arquitetura que permita agrupar os componentes de um S-RES, e definir protocolos de interação entre eles, de forma que sejam atendidos os seguintes requisitos básicos:

- A necessidade de um sistema "à prova de mudança", isto é, que limite ou atomize os impactos de mudanças no conhecimento médico em si, em medicamentos, procedimentos, legislação, etc;
- Agregação de informações de diferentes fontes e em diferente formatos;
- Existência de diversos níveis de conformidade com os padrões (seja por gradualismo ou por modularização próprios da estrutura do padrão), com impactos na modelagem de dados e na formatação de mensagens entre componentes ou entre diferentes sistemas RES;
- A necessidade de uma estratégia incremental de desenvolvimento e implantação;

No pacote AM (Archetype Model), estão os componentes relacionados aos formalismos construídos para a representação e manipulação de informação clínica. Nesse pacote encontram-se os componentes mais distintivos do OpenEHR:

1. A especificação da linguagem *Archetype Definition Language* (ADL), projetada para a formalização de modelos clínicos (estruturas de dados que representem conceitos clínicos);
2. A especificação da linguagem *Archetype Query Language* (AQL), para a busca sobre massas de dados criadas de acordo com modelos descritos em ADL;
3. A especificação da representação intermediária do modelo clínico, o padrão *Archetype Object Model* (AOM). Esse modelo de objetos determina quais são os possíveis membros ou subcomponentes de um arquétipo, e qual a relação entre esses componentes. Isto é, um objeto especificado no AOM é uma prescrição do formato da estrutura de dados que representa uma instância de um arquétipo.
4. A especificação *Operational Template Format* (OPT) apresenta o formato de arquivo projetado para descrever a representação intermediária entre modelos clínicos e formatos externos, como *eXtensible Markup Language* XML;

Seguindo a comparação feita na documentação oficial do OpenEHR, podemos dizer que um artefato ADL corresponde à maior lista possível de tipos de blocos "lego" que podem se associar para gerar um objeto de interesse (um conceito clínico). Um *template*, por sua vez, corresponderia a um subconjunto dessa lista, que resulta numa versão menor, embora consistente, do objeto definido em ADL. O artefato AOM corresponderia, então, a uma folha de instruções, indicando a melhor estratégia para encaixe entre as peças.

São essas três especificações que dão identidade ao OpenEHR. Elas são as ferramentas de "engenharia do conhecimento" e são elas que permitem o desacoplamento entre a

representação do conhecimento clínico e o aparato tecnológico dando suporte ao registro e manipulação desse conhecimento. Dessa forma, ambos, conhecimento e tecnologia, podem evoluir de maneira independente, diminuindo riscos de obsolescência, inconformidade e incompatibilidade [46].

No pacote mais básico, o *Reference Model* (RM), encontramos instâncias concretas, construídas com as ferramentas do pacote AM, combinadas aos elementos tecnológicos externos: terminologias, tipos de dados, formatos de representação, entre outros. Esse pacote traz o grupo de especificações de maior interesse para a pesquisa exposta nesse trabalho. A razão é simples: esse pacote abriga os elementos de ligação entre os modelos genéricos, criados com as linguagens do pacote AM, e as soluções tecnológicas concretas.

Este trabalho trata justamente da elaboração de uma solução de segurança baseada na natureza dos dados persistidos ou processados na nuvem. Daí a necessidade de examinar a representação física de bases de dados modeladas sob a orientação dos padrões OpenEHR. É necessário analisar a natureza e o formato dos dados persistidos para identificar quais são as operações atômicas mais recorrentes: sejam comparações textuais, comparações numéricas, agregações ou operações aritméticas.

3.5 Ciclo de vida de um RES OpenEHR

O projeto de um RES com o uso dos padrões OpenEHR tem um ciclo bastante característico. Ao contrário dos modelos clássicos, no OpenEHR, o especialista no domínio (clínico, sanitarista, pesquisador em saúde) tem participação muito mais extensa e determinante. O profissional de tecnologia participa apenas no desenvolvimento e implantação dos serviços em torno de um modelo de dados independente. A Figura 3.3 mostra uma visão panorâmica desse ciclo.

O corpo especialista produz os modelos de dados, arquétipos e *templates*, e realiza a curadoria desses modelos – revisões, correções e eventuais declinações ou extinções. A partir dos modelos básicos, de valor universal, são criadas as contextualizações, que podem atender às especificidades da legislação em um dado território, da organização interna de um dado sistema de atenção à saúde, ou até mesmo às necessidades de uma implementação específica de RES.

Com os modelos definidos e até mesmo contextualizados, o desenvolvedor é adicionado à equipe, que continua tendo os especialistas clínicos como principais atores. Nessa fase, são desenvolvidos os *templates* operacionais, que são versões "compiladas" – isto é, listas concretas de entradas de dados num formato externo (e.g. XML, JSON) – de arquétipos ou *templates*. Por isso, os *templates* operacionais são a forma mais eficiente de mapeamento

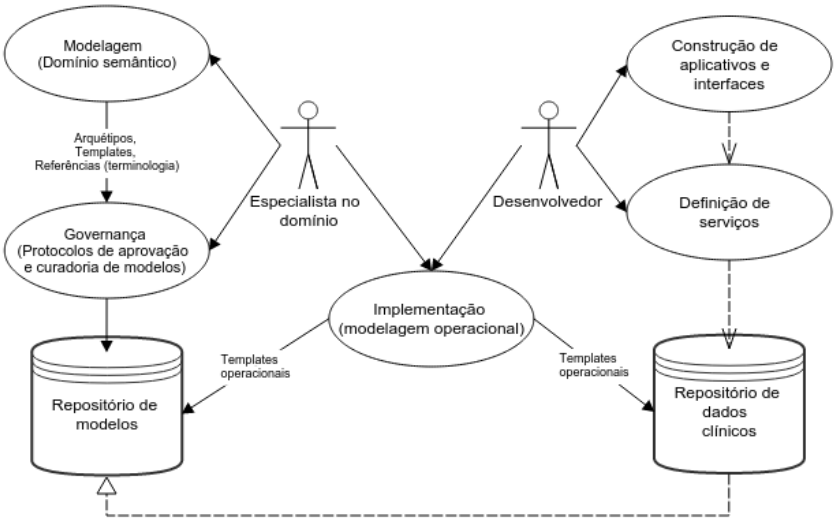

Figura 3.3: Ciclo de vida da modelagem OpenEHR

entre os modelos clínicos conceituais e os modelos físicos de dados, isto é, as instâncias persistidas nos bancos de dados.

Modelos clínicos

Os arquétipos são modelos clínicos criados por especialistas no domínio. Eles representam a agregação máxima de dados que podem contribuir para a correta informação acerca do conceito descrito no arquétipo. Os *templates*, por sua vez, são visões contextualizadas dos arquétipos, que apontam para um conjunto consistente – isto é, que não acarrete perda semântica – de entradas previstas em um ou até mesmo vários arquétipos.

Dessa forma, *templates* não são necessariamente menores que arquétipos, pois podem combinar entradas de diversos arquétipos. Formalmente, o *template* é um arquétipo, escrito em ADL ou AOM. A diferença é simplesmente semântica: enquanto o arquétipo representa um conceito clínico auto-contido (independente do contexto), o *template* representa uma informação contextualizada.

A Figura 3.4 mostra uma representação gráfica, um mapa mental, do arquétipo "Pressão sanguínea" (openEHR-EHR-OBSERVATION.blood_pressure.v1), e dá uma ideia da organização hierárquica das estruturas de dados representadas nos arquétipos.

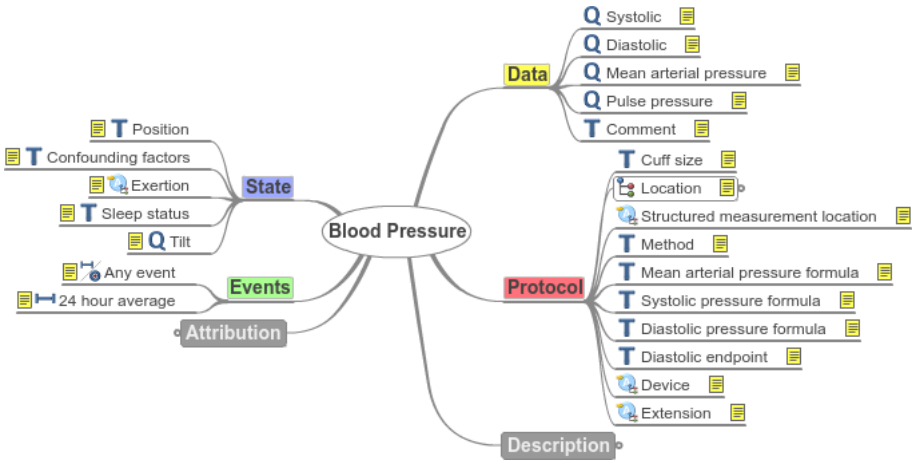

Figura 3.4: Representação gráfica de um arquétipo OpenEHR

Terminologia

O OpenEHR especifica uma codificação própria, com vistas à simplificação do referenciamento na formulação dos arquétipos. Essa terminologia, no entanto, é completamente mapeada, por construção, em padrões já amplamente difundidos, tais como SNOMED-CT e LOINC. Codificação, nesse contexto, diz respeito a um sistema de identificação unívoca para termos ou conceitos constantes em um vocabulário controlado ou ontologia. Além de prover eficiência para geração, gravação, transporte e busca sobre os registros, um vocabulário controlado permite também que a comunicação entre os geradores e consumidores da informação seja mais segura, dirimindo uma série de riscos atrelados a falhas na interpretação do relato de um profissional ou paciente [116].

3.6 Estrutura física de um RES OpenEHR

De maneira simples, pode-se definir um RES construído sob o OpenEHR como a combinação dos seguintes elementos: modelos clínicos, incluindo os *templates*, e a codificação (índice/valor) de terminologias e mapas de termos.

Um arquétipo é sempre auto-suficiente, isto é, inclui toda a informação necessária para representar um conceito clínico [43]. Os arquétipos são compostos por estruturas complexas definidas na especificação *Data Structures Information Model*. São elas:

- ITEM_SINGLE - uma entrada simples, isto é, correspondente a um tipo básico;
- ITEM_LIST - uma composição de tipos básicos;
- ITEM_TABLE - uma estrutura iterada, isto é, com várias repetições ou iterações, de uma estrutura do tipo ITEM_LIST;

- ITEM_TREE - uma lista hierarquicamente organizada de tipos básicos e outras estruturas desse pacote.

Com vistas à compatibilização com o padrão ISO/EN 13606, nos arquétipos, as estruturas do tipo ITEM_SINGLE são identificadas como um objeto da classe ELEMENT. As demais são identificadas como objetos da classe CLUSTER. O padrão ISO/EN 13606 é a norma vigente na comunidade Europeia para comunicação e interoperabilidade entre sistemas RES. A relação entre essas estruturas de dados pode ser vista na Figura 3.5.

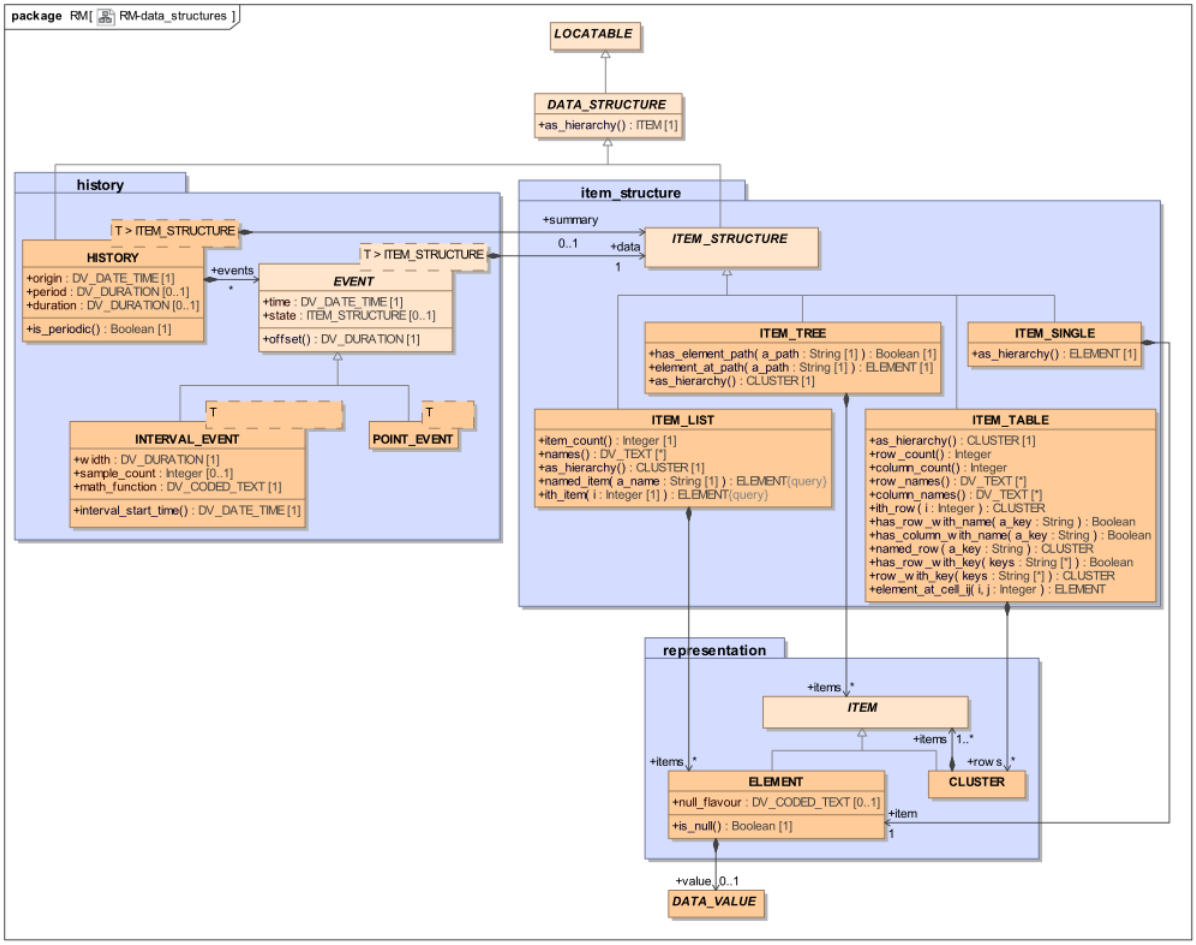

Figura 3.5: Estrutura de dados de um RES OpenEHR

Os objetos da classe ITEM_SINGLE terão sua estrutura interna definida de acordo com os tipos básicos, definidos no modelo *Data Type Information Model*. As classes definidas nesse modelo representam o menor nível, a granularidade mais fina, na definição de um arquétipo ou *template*. Esse modelo é subdividido em seis pacotes:

1. **Basic** - descreve dados boolianos, identificadores e marcadores de *status*;
2. **Quantity** - expressa quantidades, medições e intervalos;

2.1. **DateTime** - um sub-pacote, que especifica o registro de *timestamps* e datas;

3. **Time specification** - eventos iterados e duração de intervalos;
4. **Text** - texto aberto, listas/mapas de termos e texto codificado;
5. **URI** - utilizado para descrever endereços ou identificadores de recursos externos;
6. **Encapsulated** utilizado para binários, tais como imagens, vídeos, sons, bem como formatos textuais complexos (e.g. XML e HTML).

Note que o pacote *Data Types* define os tipos básicos, ou os menores elementos de dados, a serem referenciados na descrição de um arquétipo ou *template*. Mas esses tipos básicos são, na verdade, classes, ou estruturas de dados compostas pelos tipos clássicos (numérico, *string*, booliano, etc.), definidos nos padrões ISO 8601 e 11404 [63, 60].

O modelo define 54 classes distintas, com 102 atributos, no total. Algumas classes no modelo, a exemplo da classe DATA_VALUE, são abstratas. Elas compõem o modelo apenas como elementos de organização lógica do sistema de herança interno do modelo. O tipo DATA_VALUE é a raiz desse sistema de herança. Além disso, várias classes são usadas para definir atributos de outras classes no modelo. A classe CODE_PRHASE, por exemplo, figura como atributo em 10 classes. A classe DV_CODED_TEXT, por sua vez, é atributo de outras 7.

Estudando essas estruturas numa representação arbórea, e expandindo os atributos internos até chegar aos tipos clássicos (ISO 8601/11404) (c.f Figura 3.6, chega-se a 50 nós terminais possíveis. Destes, temos 1 longo, 1 numérico de dupla precisão, 3 numéricos de ponto-flutuante, 4 *timestamps* (representação numérica de data/hora), 5 inteiros, 8 booleanos e 28 *strings*. Essa primeira análise poderia levar à conclusão de que a representação física dos registros seria majoritariamente textual.

Uma análise detalhada dos arquétipos publicados no CKM, no entanto, mostrou que as classes CODE_PHRASE e CV_CODED_TEXT, do pacote 'Text', têm o maior número de ocorrências. Essas classes são seguidas, em ocorrências, por aquelas definidos no pacote 'Quantity'. A classe DV_QUANTITY, por exemplo, é usada para expressar medições e combina um valor numérico de ponto flutuante, representando a magnitude da medida, um *string* que representa a unidade de medida e, opcionalmente, um inteiro indicando precisão do ponto flutuante. As classes CODE_PHRASE e DV_CODED_TEXT, por sua vez, representam um termo dentro de uma terminologia ou ontologia – seja a ontologia embarcada no próprio arquétipo, ou uma terminologia externa. Em ambos os casos, o tipo é composto por dois elementos terminais: o código do termo, e o identificador da terminologia a que pertence. Esses dois elementos são definidos como *strings* na especificação, mas, como se tratam de elementos tomados em listas fixas, fechadas, são

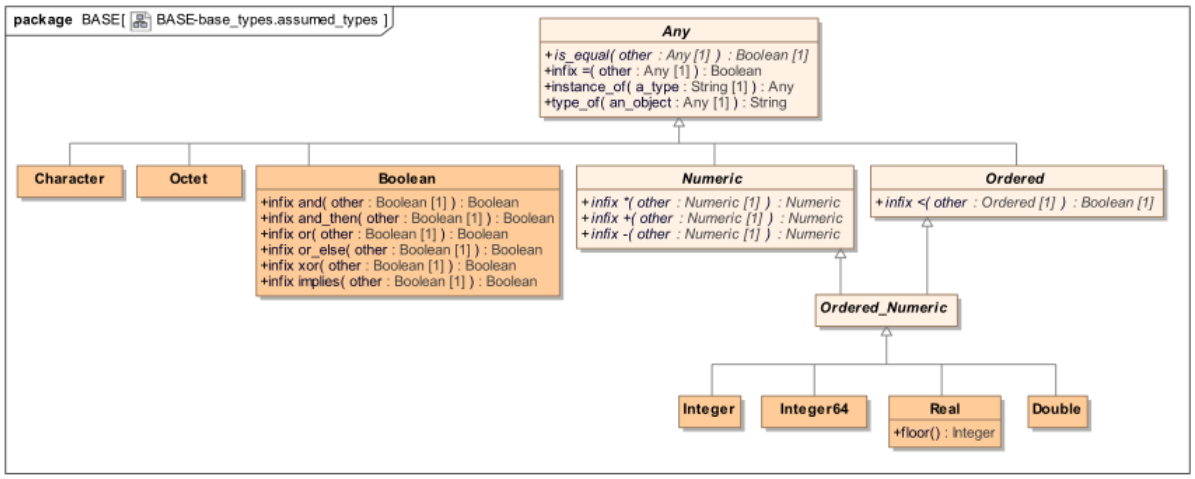

Figura 3.6: Nós terminais da representação física do arquétipo

mais eficientemente representados como índices de uma tabela global de mapeamentos de termos.

Em janeiro de 2016, o CKM público do OpenEHR contava com 404 arquétipos, dos quais apenas 47 se encontravam no *status* publicado – isto é, estavam no estágio de elaboração/revisão adequado para um sistema em produção. Nestes 47 arquétipos temos 501 entradas de tipos básicos (definidos no modelo 'Data Type'). Entre elas, 91 são da classe CODE_PHRASE, 90 DV_CODED_TEXT, 29 DV_ORDINAL, 29 DV_DURATION, 22 DV_DATE_TIME, 21 DV_QUANTITY, 21 DV_COUNT, 21 DV_BOOLEAN, 5 DV_INTERVAL, 5 DV_DATE, 5 DV_IDENTIFIER. Num banco de dados relacional, todos esses elementos de dados resultam em representações físicas de natureza numérica. Somados, representam 67% dos nós terminais dos arquétipos publicados.

Além disso, cada entrada no arquétipo, seja da classe ITEM_SINGLE ou das classes mais complexas definidos no modelo 'Data Structures', tem um identificador chamado 'node_id', que é um código mapeando aquela entrada dentro da ontologia embarcada no próprio arquétipo. A raiz do arquétipo é também o primeiro item da ontologia: é a definição do conceito modelado no arquétipo, e recebe sempre o código at0000. Os demais nós serão endereçados na ordem de inclusão no arquétipo, e recebem os identificadores at0001, at0002, (...), nos arquétipos, ou id1, id2, (...), nos *templates*. Evidentemente, esses códigos, em seu formato original, só fazem sentido dentro do arquétipo. A sua implementação em banco de dados muito provavelmente envolverá a criação de uma tabela com todos os códigos, com as respectivas definições e com uma referências ao arquétipo de origem e substituirá, nas instâncias de dados (o registros clínicos), o código alfanumérico pelo correspondente índice numérico na tabela dos 'node_id'.

Pode-se inferir que essas entradas são as mais importantes em termos de recuperabilidade do registro, isto é, em termos de identificação de registros de interesse. A probabilidade de que o médico busque por registros com um tipo específico de doença ou com um tipo específico de observação, ambos em uma terminologia fechada, codificada, é muito maior do que a de que procure por registros nos quais uma dada palavra foi escrita em um campo de texto livre. Identificadores de doenças, exames, procedimentos, remédios, entre outros, todos vêm de terminologias fechadas e são representados como um índice numa lista fixa.

3.7 Limitações do framework

As ferramentas de modelagem do OpenEHR produzem apenas artefatos estritamente atrelados ao conceito de Prontuário Eletrônico do Paciente (PEP), que inclui registros de ações, observações, avaliações, instruções e entradas administrativas (admissão, internação, etc.), além dos registros demográficos (identificação, endereços, etc.). Não existem, no entanto, soluções concretas para requisitos externos ao domínio clínico. Assim, aspectos indispensáveis a qualquer sistema informacional, tais como segurança da informação (integridade, confidencialidade e privacidade), e tolerância a falhas não são adequadamente cobertos pelos modelos OpenEHR.

O modelo EHR (da camada RM) contém um objeto EHR_ACCESS, uma instância do modelo *Security Information Model*, que trata de políticas de acesso baseadas em papéis: clínicos, enfermeiros, paciente, etc. Mas não faz qualquer conexão entre esses papéis e os arquétipos demográficos, por exemplo, que poderiam identificar quais papéis são desempenhados por cada usuário.

Um conceito essencial na área de registros eletrônicos em saúde é a anonimização do registro. Isto é, a exposição dos registros clínicos não pode ser suficiente para identificar o paciente. A resposta do OpenEHR para esse problema é bastante lacônica: os arquétipos simplesmente não trazem qualquer entrada de dados que identifique um paciente ou um prontuário específico. O atributo 'ehr_id' é definido no modelo de alto nível EHR, mas, assim como o objeto EHR_ACCESS, o modelo de alto nível não especifica nenhuma conexão direta desse atributo com os modelos clínicos em si. A maneira como essa correlação é feita também depende da implementação. Todos esses aspectos dependentes da implementação podem resultar em deficiências na capacidade de interoperabilidade do sistema.

Outra potencial vulnerabilidade decorre do modelo de desenvolvimento proposto pelo *framework*, que é bastante sensível à participação de especialistas, entre médicos e outros profissionais com domínio de informação em saúde. Se o engajamento desses profissionais

se esvair durante qualquer fase do ciclo de vida de um RES de grande porte, como aqueles desenvolvidos por autoridades sanitárias nacionais e agência centrais de saúde, toda as vantagens oferecidas pela abordagem OpenEHR podem se perder. Ou pior, pode haver uma drástica queda na razão custo/benefício da adoção dessas especificações e modelos, os quais são razoavelmente complexos.

3.7.1 Mapeamento arquétipo-relacional

Outro problema em aberto, e que merece uma análise mais detalhada, é a definição do mapeamento mais eficiente entre arquétipos e a representação física no banco de dados. A estrutura do objeto AOM apresenta o conjunto mínimo de dados para formar correta representação de uma instância de arquétipo em memória. No entanto, a transformação dessa estrutura em memória para uma estrutura eficiente, do ponto de vista da velocidade de busca e do custo de armazenamento, ainda é um desafio.

Uma pesquisa realizada para o governo da província de Victorya na, Austrália, chegou à conclusão de que não era possível definir um mapeamento formal entre um modelo OpenEHR e um modelo relacional [83]. A conclusão desses pesquisadores foi a de que o OpenEHR produz um modelo lógico que pode ser implementado de diversas formas, quando transposto para um modelo físico. Isto é, muitos elementos de lógica do negócio e manipulação dos dados dependerão da implementação.

Uma abordagem mais simples seria utilizar um banco de dados orientado a objetos, que persistisse estruturas no mesmo formato em que são manipuladas em memória. Mas as vantagens dessa opção se limitam às operações de gravação e leitura. Estudos específicos sobre o armazenamento de modelos OpenEHR nesse tipo de banco apontam para uma *performance* muito pobre em operações cruciais como busca e exportação de dados, o que inviabiliza seu uso em sistemas maiores [84].

Diante desse desafio, duas alternativas foram consideradas: construir uma solução genérica, que gerencie em tempo real o mapeamento de todas as cadeias de entradas de arquétipos e *templates*, e as referências entre esses elementos; ou construir um modelo físico a partir de um *template* operacional – uma estrutura "engessada", representando apenas o conjunto de dados útil no contexto de uma aplicação específica.

A primeira opção é extremamente complexa e demanda um esforço de engenharia e programação que fugiria em muito do escopo deste trabalho. A segunda opção é limitada a um número reduzido de arquétipos e, mais especificamente, a uma versão específica deles. Uma vez que o objetivo da construção do Safe-Record é apresentar uma solução viável de segurança, e não necessariamente um produto OpenEHR comercializável ou pronto para produção, foi adotada a estratégia mais simples.

A Figura 3.7 apresenta uma representação lógica, simplificada, do modelo efetivamente persistido na nuvem.

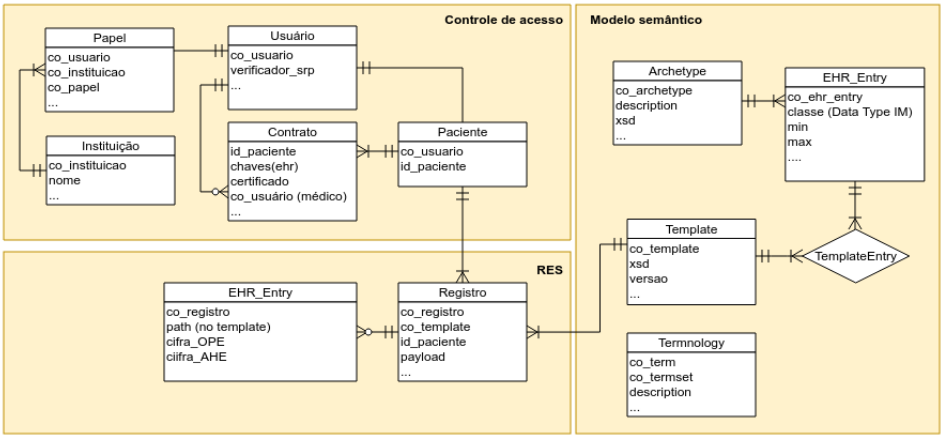

Figura 3.7: Modelo relacional lógico

3.8 Considerações finais

Apesar das limitações, o OpenEHR representa uma tendência importante no contexto da informática em saúde. Todo o trabalho da comunidade OpenEHR mostra que os profissionais da saúde sentem a necessidade de uma participação mais determinante no projeto dos sistemas dando suporte às suas atividades. Isso mostra que o respeito a restrições semânticas desse domínio é indispensável. Qualquer proposta de segurança de aplicações RES deve, portanto, passar primeiramente pela elicitação das necessidades reais dos profissionais da área. Somente depois de ter atendido às restrições semânticas dos componentes de software e modelos de dados resultantes é que uma solução de segurança pode ser considerada como bem sucedida ou viável.

O estudo dos padrões OpenEHR, em especial os modelos *Data Structures* e *Data Types* do pacote RM, mostram que os sistemas projetados em conformidade com o OpenEHR são bastante complexos. A representação física dos dados é bastante complexa. São diversas estruturas de dados flexíveis e interdependentes, com subcomponentes intercambiáveis. Uma análise bastante detida dos arquétipos publicados no CKM mostrou que no limite inferior de granularidade, isto é, nos nós terminais dessas estruturas, a maior parte das entradas de dados terá natureza numérica.

Dessa análise decorre a conclusão de que a busca e as demais operações (comparações, medidas de dispersão, etc.) realizadas sobre os RES se dará por meio de uma enorme quantidade de operações numéricas: especialmente comparações de ordem, somas e multi-

plicações. Essa conclusão orientou a continuidade deste trabalho de pesquisa na busca por criptossistemas que permitissem operações numéricas básicas sobre os dados encriptados, de maneira que, mesmo delegando a computação para o servidor na nuvem, o consumidor não expusesse qualquer informação sensível dos pacientes.

4 Mecanismos criptográficos selecionados

Neste capítulo são apresentados os esquemas e primitivas criptográficos utilizados no SafeRecord, bem como as características que justificam sua seleção. Como mostra a Tabela 4.1, eles estão agrupados neste capítulo de acordo com sua aplicação: o primeiro grupo, com esquemas bem consolidados e com larga aplicação na indústria e na literatura, é utilizado no protocolo de gestão de chaves que garante o controle de acesso aos dados. O segundo, que contém alguns esquemas menos difundidos, se presta à execução de operações sobre a massa de dados encriptados na nuvem, sem que o conteúdo dos dados seja revelado a um atacante com acesso à nuvem. A Figura 4.1 mostra as estratégias de emprego de cada mecanismo.

Tabela 4.1: Sumário das primitivas criptográficas

Primitiva	**Emprego**
	Controle de acesso
SRP	Protocolo para autenticação por prova de conhecimento
RSA-OAEP + PBKDF2	PBE utilizado para guarda das chaves de um usuário
RSA-SSP	Primitiva de assinatura digital usada nos processos de autenticação e autorização
	Manipulação dos registros em saúde
AES	Cifra principal, guarda todos os elementos dos registros clínicos
AHE (Paillier)	Cifra aditivamente homomórfica, usada para somas na nuvem
OPE (Boldyreva)	Cifra com preservação de ordem, usada para pesquisas na nuvem
DSE	AES determinístico, cifra executada sobre identificadores

Os esquemas criptográficos foram escolhidos em função de sua adequação aos requisitos funcionais mínimos do domínio da aplicação: as operações que cada ator precisa executar sobre os dados. Também foram considerados os seguintes requisitos não funcionais:

- O desafio de autenticação de usuário gravado na nuvem não pode ser usado para recuperação da senha ou roubo da identidade do usuário;

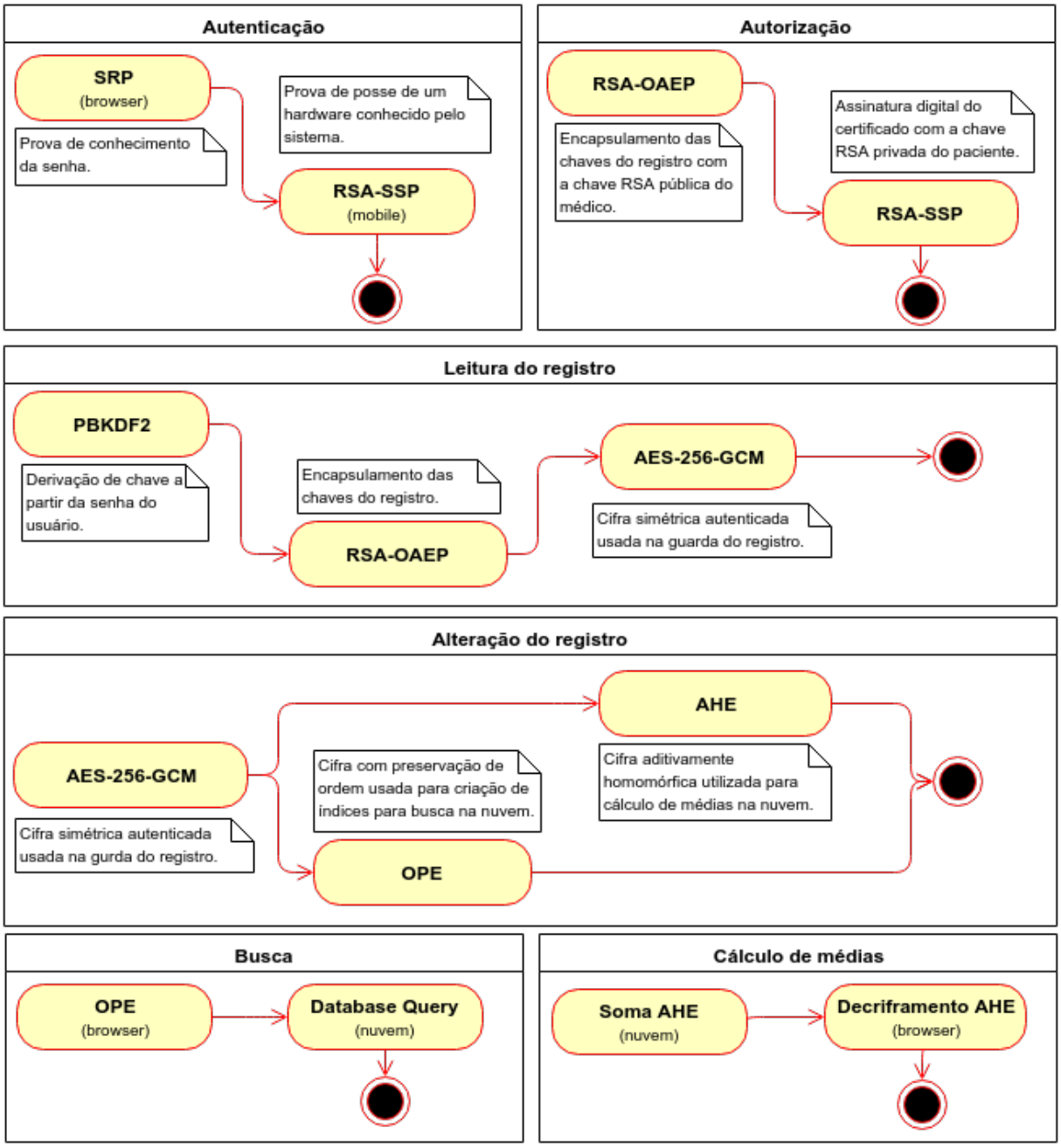

Figura 4.1: Estratégia de emprego dos mecanismos criptográficos

- A autenticação não pode se limitar a um desafio relacionado à senha, pois esta pode ser encontrada por meios estranhos ao sistema, especialmente quando o usuário usa a mesma senha em diversas aplicações;
- O protocolo de autenticação não pode ser utilizado como vetor de ataque contra o sistema;

- O mecanismo de controle de acesso deve ser simples, eficiente e aplicável a qualquer serviço de nuvem;
- A autorização de acesso à informação clínica deve ser imposta por uma primitiva criptográfica: isto é, apenas os portadores da chaves criptográficas adequadas pode ler ou alterar o registro;
- A autorização não pode ser forjada nem refutada;
- O mecanismo de controle de acesso não pode inviabilizar o acesso emergencial ao prontuário do paciente;
- Todas as operações de cifragem e decifragem devem ocorrer na máquina do usuário;
- O sistema deve permitir a comparação numérica (buscas por intervalos) sobre as cifras na nuvem;
- O sistema deve permitir a operação de a soma sobre as cifras na nuvem;
- As operações de busca, *download*, decifragem e exibição do registro em tela não podem demandar um tempo de processamento e espera que degrade ou inviabilize a usabilidade do sistema.

Dessa forma, criptossistemas que exigissem um alto nível de interatividade ou proficiência tecnológica por parte dos usuários (pacientes ou profissionais de saúde), não poderiam ser considerados. Qualquer solução de autenticação e autorização, no contexto médico, deve ser condicionada a esses requisitos. Afinal, um médico não pode esperar que um paciente em choque anafilático digite a senha que dá acesso a seu prontuário, ou que um banhista afogado porte o *token* criptográfico que decripta seus dados clínicos.

Ademais, é necessário considerar o ordenamento jurídico que regula a informação clínica que, via de regra, recai sobre a instituição prestadora de serviços de auxílio à saúde (IAS) [29, 58]. No Brasil, por exemplo, a Resolução 1.638/2002 do Conselho Federal de Medicina, que defini o Prontuário Médico e dá providências, determina que o prontuário sofra revisão periódica por uma junta médica, como forma de regulação ou controle interno do ato médico. Essa junta é eleita pelo corpo clínico da IAS para um determinado período, de forma que o paciente pode sempre ser informado a respeito de quem tem acesso a seu prontuário - além de seu médico. Requisitos semelhantes existem em praticamente todos os países.

4.1 Controle de acesso

Os esquemas propostos para o controle de acesso são sistemas maduros, bastante difundidos e presentes em qualquer biblioteca ou API de recursos criptográficos. A imposição de tais característica deriva do fato de que o controle de acesso deve ser simples, eficiente e aplicável a qualquer tipo de serviço de nuvem. Não pode depender, portanto, de configurações do sistema operacional, nem da presença de serviços, APIs ou bibliotecas específicos ou pouco difundidos.

O controle de acesso do Safe-Record faz uso de um protocolo de autenticação por dois fatores, sendo que o primeiro estágio usa uma primitiva criptográfica para comprovação de senha e o segundo estágio utiliza assinaturas digitais para comprovação de posso de um equipamento conhecido pelo sistema.

O projeto também inclui um sistema de criptografia assimétrica, usado na produção dos certificados e assinaturas digitais que atestam os direitos de acesso de cada usuário. O sistema assimétrico também é usado no encapsulamento das chaves simétricas que dão acesso aos registros. Os sistemas escolhidos são detalhados a seguir.

4.1.1 SRP

Aplicações *web* costumeiramente guardam as senhas dos usuários na forma de um desafio de autenticação produzido com uma função *hash* de propósito geral. O *hash* é aplicado sobre uma combinação da senha do usuário com uma sequência aleatória, selecionada durante seu registro. Em aplicações um pouco mais seguras, em lugar do *hash* de propósito geral, utiliza-se algoritmos de custo adaptativo, como o Bcrypt, que é baseado numa cifra de bloco [33].

O problema de armazenar um *hash* da senha é que essas funções são muito rápidas. De acordo com o trabalho de Provos e Mazières [97], o baixo custo computacional de funções de *hash* de propósito geral, como MD5, SHA1 ou SHA256, as torna inseguras contra ataques de dicionário. Em 1999 já era possível testar em média 200.000 entradas por segundo numa máquina comum, e num servidor especializado, usando computação paralela em GPUs, até 700 milhões por segundo.

O algoritmo Bcrypt foi proposto por esses autores como uma alternativa de defesa adaptativa. Isto é, a implementações dos procedimentos de registro e verificação da senha pode ser configurada para aumentar gradativamente o custo de execução do algoritmo, de forma a acompanhar a evolução do poder computacional do atacante médio. Em 1999, os autores acreditavam que um custo 5 ordens de magnitude maior que o do SHA1, por exemplo, era suficiente para o cenário da época.

Com o advento da computação em nuvem, em que um atacante tem acesso a um enorme poder computacional, o custo de um ataque de força bruta tornou-se praticamente insignificante. A corrida por aumentar a dificuldade do desafio perde o significado, pois um nível de custo realmente inviável para o atacante também seria inviável para o usuário legítimo.

Se a verificação de identidade baseada em senha/verificador for um requisito, a resposta mais adequada é procurar por um protocolo que utilize primitivas mais fortes e que inviabilize definitivamente a reversão da senha a partir do verificador armazenado na nuvem. Por outro lado, o protocolo deve garantir que o custo para executar o protocolo de verificação seja razoavelmente baixo – para que essa operação não se torne um vetor para ataques de negação de serviço contra o servidor.

O protocolo *Secure Remote Protocol* (SRP), proposto por Tom Wu, da Universidade de Stanford, atende a esses requisitos [120]. Esse protocolo une a verificação da senha ao estabelecimento de uma chave criptográfica para segurança da comunicação entre cliente e o servidor no decurso de uma sessão. Ao fim da execução do protocolo, ambos terão uma chave semelhante àquela gerada num protocolo Diffie-Hellman, derivada a partir da senha, no lado do cliente, e do verificador, no servidor. A chave é válida apenas naquela sessão, pois deriva tanto de um segredo previamente compartilhado (por meio do verificador), bem como de números aleatoriamente selecionados pelo cliente e pelo servidor para aquela sessão.

Neste trabalho de pesquisa, adotou-se uma implementada da versão SRP-6a do protocolo [121], conforme a descrição da RFC 5054 [111]. Para o protocolo SRP-6a a definição de verificador é dada por:

$$\mathrm{x} = \mathrm{H}(\mathrm{S}||\mathrm{H}(\mathrm{id}||\mathrm{P})) \tag{4.1}$$

$$\mathrm{v} = \mathrm{g}^{\mathrm{x}} \mod \mathrm{N} \tag{4.2}$$

onde,

v : verificador da senha P;

P : senha do usuário;

id : identificador do usuário;

S : sequência selecionada aleatoriamente no ato do registro;

x : chave privada, derivada de id, P e S;

g : um gerador do grupo multiplicativo $\mathbb{Z}_{\mathrm{N}}$;

N : um primo na forma $(2q + 1)$;

q : um primo de Sophie Germain grande (1024 bits).

A figura 4.2 mostra as mensagens trocadas entre o cliente e o servidor durante a execução do protocolo. Note que todas as operações são executadas sobre elementos do grupo multiplicativo $\mathbb{Z}_N$, e, portanto, em módulo N:

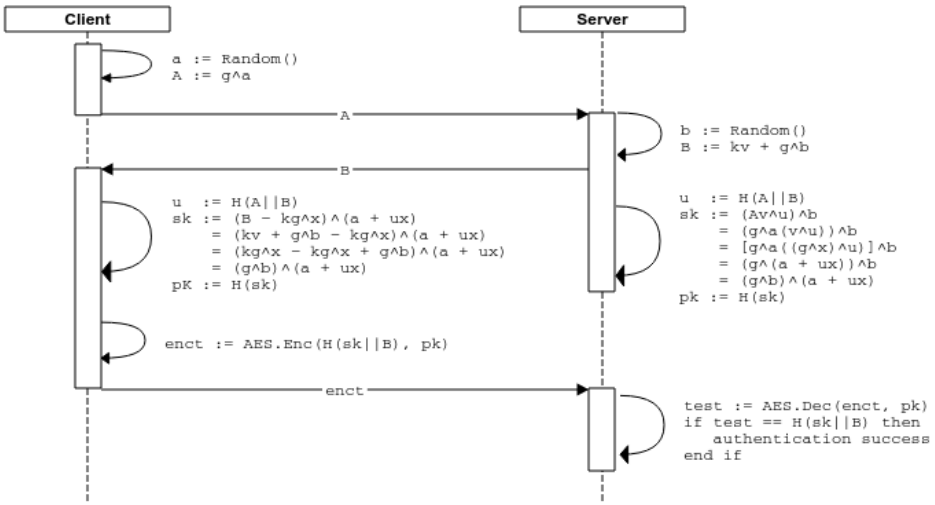

Figura 4.2: Execução do protocolo SRP

4.1.2 PBKDF2

A função PBKDF2 (Password-Based Key Derivation Function v2.0), definida no padrão PKCS#5 v2.0 (RFC 2898 [68]) consiste na aplicação iterada de uma permutação pseudo-aleatória sobre uma combinação da senha e de uma sequência de *salting*. Essa sequência, geralmente, é gerada no servidor, durante o registro do usuário, e é suprida ao cliente durante o processo de autenticação.

A função resultante tem o seguinte formato:

$$\mathrm{DK} = \mathrm{PBKDF2}(\mathrm{PRF}, \mathrm{P}, \mathrm{S}, \mathrm{i}, \mathrm{l})$$

Onde:

DK chave derivada;

PRF função pseudo-aleatória escolhida;

P senha do usuário;

S sequência aleatória;

i número de iterações da PRF;

l comprimento (em bytes) requerido para a chave derivada;

A função de permutação adotada na implementação do Safe-Record é um HMAC construído com a função de *hash* da família SHA1. São realizadas cem iterações para derivar uma chave de 32 bytes. A sequência de *salting* utilizada é a mesma que é gerada no contexto do protocolo SRP, durante o registro do usuário.

4.1.3 RSA-OAEP

O protocolo de controle de acesso projetado para este trabalho faz uso de um esquema criptográfico assimétrico para a cifra do conjunto de chaves criptográficas utilizadas em outras funcionalidades da aplicação. Esse protocolo também faz uso de assinaturas digitais para validar as autorizações de acesso criadas por cada usuário.

Diante desses requisitos funcionais - cifra assimétrica e assinatura digital - e dos não funcionais - maturidade, disponibilidade em bibliotecas padrão e facilidade de aplicação - o criptossistema mais adequado encontrado foi o RSA. Esse sistema pode ser descrito, basicamente, como a combinação de três algoritmos:

RSA.KeyGen: o algoritmo de geração de chaves, primeiramente, encontra dois primos grandes (com representação digital, por exemplo, de 2048 ou 4096 bits) p e q, computando também seu produto n = pq. O algoritmo então seleciona um natural e tal que $\text{mdc}(e, \phi(n)) = 1$, onde $\phi(n)$ é a função totiente em n, i.e. $\phi(n) = (p-1)(q-1)$. Por fim, o algoritmo computa $d = e^{-1} \mod \phi(n)$. A chave pública será representada pela tupla $\langle n, e \rangle$, e a privada por $\langle n, d \rangle$;

RSA.Enc: o algoritmo de cifra consiste computar $c = m^e \mod n$, dadas a chave pública $\langle n, e \rangle$ e um texto em claro $m \in \mathcal{M}$ - onde $\mathcal{M}$ é o espaço de mensagens;

RSA.Dec: o algoritmo de decifração recebe como entradas a chave privada $\langle n, d \rangle$ e uma cifra c, computando a mensagem correspondente da seguinte forma: $m = c^d \mod n$.

Note que a segurança desse criptossistema se baseia na dificuldade encontrar a e-ésima raiz módulo n do inteiro c. Este problema, introduzido no trabalho de Rivest, Shamir e Adleman, acabou conhecido como 'Problema RSA' e corresponde, em dificuldade, ao problema da fatoração de números compostos por primos grandes (uma vez que é custoso encontrar tais primos). Até hoje, nenhum algoritmo proposto resolve este problema em tempo polinomial [99].

Essa variante mais básica do sistema RSA, no entanto, não tem a propriedade de segurança semântica. Isto é, não é segura contra ataques de texto escolhido (CPA), uma vez que a função de cifra é determinística: não adiciona qualquer elemento de aleatoriedade que impeça um atacante de tomar conclusões relevantes ao comparar dois textos cifrados. Por isso a variante selecionada nesta pesquisa foi o RSAES-OAEP[1] (*RSA Encryption System - Optimal Asymmetric Encryption Padding*), definida no padrão PKCS#1 v2.1 (RFC 3447 [67]), que utiliza uma técnica de *padding* (preenchimento) prévio da mensagem m com um número de bits tomados de uma fonte de aleatoriedade – de acordo com o parâmetro de segurança da chave. Essa técnica torna o esquema seguro contra ataques do tipo CPA e CCA (*chosen-cyphertext attack*) [12].

Como consequência da magnitude dos números utilizados no RSA, tanto a computação da cifra quanto da assinatura são bastante custosas. Por isso, a maior parte das aplicações práticas utiliza esse sistema apenas para a cifra de uma chave a ser utilizada em um sistema simétrico, e não para cifra do dado em si. Uma vez que compartilham a chave simétrica, os participantes da comunicação passam a trocar mensagens cifradas com um algoritmo muito mais eficiente. A técnica empregada neste trabalho é a do RSA-KEM, definida na RFC 5990 [98], que consiste nos seguintes passos:

- Selecionar um primo menor que o módulo n da chave pública (i.e. $\mathrm{p} \in \mathcal{Z}_\mathrm{n}$);
- Utilizar o primo p como entrada da função derivadora de chaves PBKDF2 para gerar uma chave k de 32 bits ($\mathrm{k} = \mathrm{PBKDF2}(\mathrm{p}, \mathrm{salt})$);
- Cifrar a mensagem com uma cifra simétrica eficiente (e.g. AES-256-CBC), utilizando a chave k.
- Cifrar o primo p com o esquema RSA-OAEP;

O texto cifrado resultante deste método é a combinação da cifra AES e a chave encapsulada com RSA. O detentor da chave RSA privada correspondente à chave pública utilizada no encapsulamento, ao receber esse texto cifrado, deve decifrar o primo p, derivar a chave k e então decifrar o *payload* da mensagem. No restante deste texto, para simplificação da leitura, utiliza-se apenas a expressão cifra RSA-OAEP, mas, a depender do tamanho do objeto sendo cifrado, aplica-se o método RSA-KEM.

4.1.4 RSA-PSS

A importância histórica do sistema RSA também advém do fato de que, além de prover primitivas fortes de encritação, ele dispõe de primitivas de assinatura digital. Dois algoritmos podem ser facilmente definidos:

[1] Por simplicidade, referenciado por RSA-OAEP no restante do trabalho

RSA.Sign: o algoritmo de assinatura recebe a chave privada $\langle n, d \rangle$ e uma mensagem m para criar uma assinatura $s = m^d \mod n$.

RSA.Verify: o algoritmo de verificação permite que, de posse da chave pública $\langle n, e \rangle$ e da assinatura s, qualquer pessoal seja capaz de verificar a correspondência entre s e m. A verificação consiste simplesmente em elevar s à potência e módulo n. Ou seja, $m = s^e \mod n$.

A assinatura RSA é não forjável e não repudiável. Isto é, para forjar uma assinatura RSA, o atacante precisa, primeiramente, resolver o problema RSA para encontrar o expoente privado d ou o problema da fatoração de n. No melhor conhecimento deste pesquisador, ainda não existe qualquer algoritmo eficiente (de tempo polinomial em relação ao tamanho da entrada) que resolva este problema. Pelo mesmo motivo, uma vez verificada a correspondência entre s e m, o portador da chave privada só poderá alegar que não criou a assinatura s se puder provar que outra pessoa teve acesso à chave privada.

Mais uma vez, assim como a utilização do método KEM, a magnitude dos número utilizados e o custo computacional das operações associadas faz com que a assinatura não seja executada sobre toda a mensagem, mas sim sobre um *hash* da mesma. O verificador da assinatura computa, primeiramente, o *hash* para, então, verificar se a assinatura corresponde ao resultado encontrado.

Foram selecionados os algoritmos de assinatura e verificação do esquema RSASSA-PSS [2] (*RSA Signature Scheme with Appendix - Probabilistic Signature Scheme*) definidos no padrão PKCS#1 v2.1 (RFC 3447 [67]). A versão mais recente do padrão PKCS#1 (v2.2), ainda não publicada como RFC, utiliza uma função SHA224 ou superior, com vistas à criação de *hashs* mais fortes. O esquema RSA-PSS também foi selecionado por contar com maior número de implementações, ao mesmo tempo em que oferece um nível de segurança, quanto à assinatura digital, adequado à aplicação em tela.

4.2 Manipulação de registros encriptados na nuvem

Para a manipulação dos registros encriptados na nuvem foi adotada a seguinte estratégia: o registro é armazenado com o uso de uma cifra simétrica autenticada, cujo objetivo é garantir a confidencialidade, integridade e autenticidade da informação clínica ali representada. Outra característica primordial que se buscou foi a velocidade de cifragem e decifragem, necessários para agilizar a escrita e a leitura das entradas do RES.

Para a realização de buscas e para a contagem de ocorrências de registros de um determinado tipo, foi selecionado um sistema de criptografia com preservação de ordem

[2] Por simplicidade, referenciado como RSA-PSS no restante do trabalho

(OPE, do inglês *Order-Preserving Encryption*. São cifradas com esse sistema, por exemplo, as entrada da classe Indentity, do pacote Basic e outras entradas de tipos definidos nos pacotes Quantity e Text do modelo *Data Type Information Model* do OpenEHR.

Por fim, foi selecionado um esquema aditivamente homomórfico, para permitir o cálculo de médias de valores de entradas específicas. Apenas entradas dos tipos definidos nos pacotes Quantity, incluso o subpacote DateTime, do modelo de tipos de dados do OpenEHR (detalhado no Capítulo 3) são cifradas com esse sistema.

O sistema simétrico escolhido é o AES-256-GCM, descrito a seguir, que será utilizado primariamente para agilizar a leitura do registro. As características desse sistema são especialmente relevantes para o paciente, pois, de certa forma, será o único sistema impactando o desempenho do aplicativo durante a leitura de seu registro. Para o clínico, que precisa realizar buscas sobre os registros no sistema a *performance* do AES é um elemento secundário. Sua experiência é mais influenciada pelo desempenho dos outros sistemas. Uma vez que o clínico selecione um registro ou um grupo de registros, no entanto, é a cifra AES, que tem uma função de decifragem rápida, que é utilizada para a exibição dos dados.

4.2.1 AES-256-GCM

O sistema AES (*Advanced Encryption Standard*) é o algorítimo de cifra simétrica padrão para a comunicação entre órgãos governamentais dos EUA, estabelecido pelo *National Institute of Standards and Technology* (NIST), o órgão oficial do padronização daquele país, em 2001. Trata-se de uma variante do sistema Rijndael, proposto por Joan Daemen and Vincent Rijmen, que foi vencedor do concurso lançado pelo NIST em 1997 para definir um substituto para o sistema DES [31]. O AES é também uma das cifras de bloco do padrão ISO/IEC 18033-3 [62]. Devido a sua importância como padrão de comunicação, pode-se afirmar que o AES é parte de qualquer biblioteca criptográfica atual.

O sistema Rijndael consiste em um circuito do tipo SPN (*Substitution-Permutation Network*) com tamanho variável entre 128 e 256 bits (em múltiplos de 32). O padrão AES fixa o bloco em 128 bits, e pode operar com três tamanhos de chave: 128, 192 e 256 bits. O tamanho da chave determina o número de rodadas a serem executadas no circuito. Para a chave de 256 bits (AES-256), são executadas 14 rodadas [88].

O sistema pode ser descrito a partir das seguintes rotinas:

Setup: A entrada do circuito é disposta numa matriz 4x4 de bytes, dispostos em colunas primeiro, que é chamada de matriz de estado (*StateMatrix*).

Então, uma subchave de 128 bits é derivada para cada rodada, utilizando o algoritmo de *key scheduling* (derivação cíclica de chave) KeyExpantion() de Rijndael.

Rodada inicial: a rodada inicial, consiste apenas na execução da sub-rotina **AddRoundKey**, usada para adicionar o conteúdo do bloco à subchave gerada para a primeira rodada. Essa sub-rotina é definida como uma caixa de mistura, na qual cada byte da matriz de estado é combinado com um bloco da subchave da rodada por meio de um xor bit-a-bit.

Rodadas intermediárias: em cada uma das próximas 13 rodadas, as seguintes sub-rotinas são executadas:

SubBytes uma caixa de substituição, na qual cada byte na matriz de estado é substituído de acordo com uma tabela fixa (uma matriz de substituição de dimensões iguais às de estado);

ShiftRows uma caixa de substituição, na qual as três últimas linhas da matriz de estado são a transpostas ciclicamente um dado número de vezes;

MixColumns uma caixa de mistura, que combina os quatro bytes de cada coluna;

AddRoundKey definido acima.

Rodada final: semelhante às rodadas intermediárias, com exceção da sub-rotina MixColumns, que não é executada na rodada final.

O algoritmo 4.1 dá uma visão mais dinâmica dessas rotinas:

Para cifrar uma entrada menor que o tamanho da entrada do circuito, a solução trivial é preencher a sequência de bits com zeros. A segurança da cifra não pode ser afetada por esse padrão de preenchimento. Para cifrar uma sequência maior que o circuito, a primeira providência é dividi-la em blocos de tamanho igual ao da cifra, preenchendo a última parte, caso necessário. A maneira como a cifra será utilizada para cifrar esses blocos é chamada de 'modo de operação' da cifra e afeta drasticamente a segurança do sistema resultante.

O modo de operação mais trivial, conhecido como *Electronic CodeBook* ou ECB, simplesmente cifra cada bloco individualmente. Esse modo de operação é bastante inseguro, pois a cifra de bloco é uma função determinística e a cifragem de blocos iguais com a mesma chave resulta em resultados iguais, o que revela padrões importantes no texto plano e permite ataques de criptoanálise triviais. No modo CBC (*Cipher Block Chainning*), o primeiro bloco é misturado a um vetor de inicialização (IV), com xor bit-a-bit, e o resultado da cifra do primeiro bloco é utilizado como IV para o bloco seguinte, e assim por diante. Esse modo de operação é mais seguro que o modo ECB, desde que o IV introduzido no primeiro passo tenha entropia alta.

O modo de operação escolhido para este trabalho é o GCM (Galois/Counter Mode), uma extensão do modo CTR (*Counter Mode*), introduzido por Diffie e Hellman [34], que

Algoritmo 4.1. Rijndael

Require: fixed SubstituionMatrix

```
1: Nk ← 8                                                  ▷ Número de palavras de 32 bits na chave
2: Nb ← 4                                                  ▷ Número de palavras de 32 bits na matriz de estado
3: Nr ← 14                                                 ▷ Número de rodadas
4: function RIJNDAEL(StateMatrix, CipherKey)
5:     subkeys ← KeyExpantion(CipherKey)
6:     StateMatrix ← AddRoundKey(StateMatrix, subkeys[0])  ▷ Rodada inicial
7:     for (i = 1; i < Nr; i + +) do
8:         StateMatrix ← SubBytes(StateMatrix, SubstituionMatrix)
9:         StateMatrix ← ShifRows(StateMatrix)
10:        if i < (Nr − 1) then
11:            StateMatrix ← MixColumns(StateMatrix)       ▷ Não é executada na rodada final
12:        end if
13:        StateMatrix ← AddRoundKey(StateMatrix, subkeys[i])
14:    end for
15:
16:    return StateMatrix
17: end function
18:
19: function KEYEXPANTION(Key)
20:    for i ← 0, (Nk − 1) do
21:        W[i] ← [Key[4*i], Key[4*i+1], Key[4*i+2], Key[4*i+3]];
22:    end for
23:    for i = 4; i < Nb * (Nr + 1); i++ do
24:        temp ← W[i - 1];
25:        if i  mod Nk = 0 then
26:            temp ← SubByte(RotByte(temp)) ∧ Rcon[i / Nk];
27:        else if i  mod Nk = 4 then
28:            temp ← SubByte(temp)
29:        end if
30:        W[i] ← W[i - Nk] ∧ temp;
31:    end for
32:
33:    return W
34: end function
```

transforma a cifra de bloco numa cifra de fluxo com uma técnica de *keystream* baseada no sucessivo ciframento da combinação entre o vetor de inicialização e um contador. O CTR diminui significativamente os riscos introduzidos pela necessidade de produzir um IV forte para cada mensagem cifrada, mas deve ser implementado com cuidado, pois o *nonce* que substitui o vetor de inicialização, embora não necessite de entropia alta, jamais pode ser repetido, sob pena de enfraquecer significativamente a segurança do sistema [51].

O modo GCM adiciona à segurança do modo CTR uma função de autenticação baseada numa multiplicação sobre o campo de Galois executada a cada iteração, logo após o XOR do bloco de mensagem com o bloco de chave. A combinação dessas funções resulta num sistema robusto de criptografia autenticada. Esse modo tem outra vantagem específica, que é a capacidade de paralelização da operação de deciframento, o que o torna um sistema ideal para a leitura de grandes arquivos.

Para suprir a necessidade de geração de um *nonce* na implementação deste trabalho, foi adotada uma função sobre o identificador unívoco dos registros na tabela relacional onde são gravados. Note que a segurança do sistema só depende da unicidade do *nonce*, que é público.

4.2.2 Criptografia homomórfica

Criptografia homomórfica diz respeito a um conjunto de técnicas por meio das quais é possível gerar cifras que preservem propriedades específicas de suas respectivas mensagens. Uma maneira de formalizar esse homomorfismo entre mensagem e cifra é mostrar o homomorfismo entre a imagem de uma função $\mathrm{f}_{\mathcal{M}}$ operando sobre o espaço $\mathcal{M}$ de todas as mensagens possíveis, e a imagem da uma função $\mathrm{f}_{\mathcal{C}}$, operando sobre $\mathcal{C}$, o espaço de todas as cifras possíveis.

$$\forall \mathrm{m} \in \mathcal{M}, \forall \mathrm{c} \in \mathcal{C}, \tag{4.3}$$

$$\mathrm{c} = \mathrm{Enc}(\mathrm{m}) \Rightarrow \mathrm{f}_{\mathcal{C}}(\mathrm{c}) \equiv \mathrm{Enc}(\mathrm{f}_{\mathcal{M}}(\mathrm{m})) \tag{4.4}$$

Um esquema é considerado plenamente homomórfico se esta propriedade for válida para qualquer função definida no espaço de mensagens. Isto é, $\mathrm{Enc} : \mathcal{M} \rightarrow \mathcal{C}$ é plenamente homomórfica se:

$$\forall \mathrm{f}_{\mathcal{M}} : \mathcal{M} \rightarrow \mathcal{M}, \tag{4.5}$$

$$\exists \mathrm{f}_{\mathcal{C}} : \mathcal{C} \rightarrow \mathcal{C} | \mathrm{f}_{\mathcal{C}}(\mathrm{Enc}(\mathrm{m})) \equiv \mathrm{Enc}(\mathrm{f}_{\mathcal{M}}(\mathrm{m})) \tag{4.6}$$

Considerados, por muitos anos, o 'santo graal' da criptografia, esquemas plenamente homomórficos só viram avanços significativos a partir do trabalho de Gentry [48]. Sua

tese provou, pela primeira vez, que é possível construir um sistema com homomorfismo pleno, ou seja, com o qual seja possível executar funções arbitrárias sobre as cifras. A construção de Gentry tem, no entanto, várias limitações práticas. A mais marcante entre elas é o fato de que as cifras crescem exponencialmente em relação ao tamanho do texto plano e ao número de operações que se deseja realizar sobre elas.

Um esquema plenamente homomórfico (FHE, do inglês *Fully homomorphic Encryption*) é especialmente interessante para soluções envolvendo computação em nuvem, pois permite a delegação de computações arbitrárias para terceiros, sem os perigos de ceder os dados em claro. Desde que provadamente seguro contra ataques CPA e CCA, um sistema com essa propriedade claramente vence qualquer preocupação com a segurança da informação.

Na maior parte da construções atuais, além das primitivas de ciframento e deciframento, os sistemas FHE oferecem também uma primitiva de avaliação de funções homomórficas. Também é comum que um algoritmo de derivação de chaves específicas para essa primitiva, além das chaves pública e privada (ou de ciframento e deciframento). A Figura 4.3 dá uma ideia dessas construções.

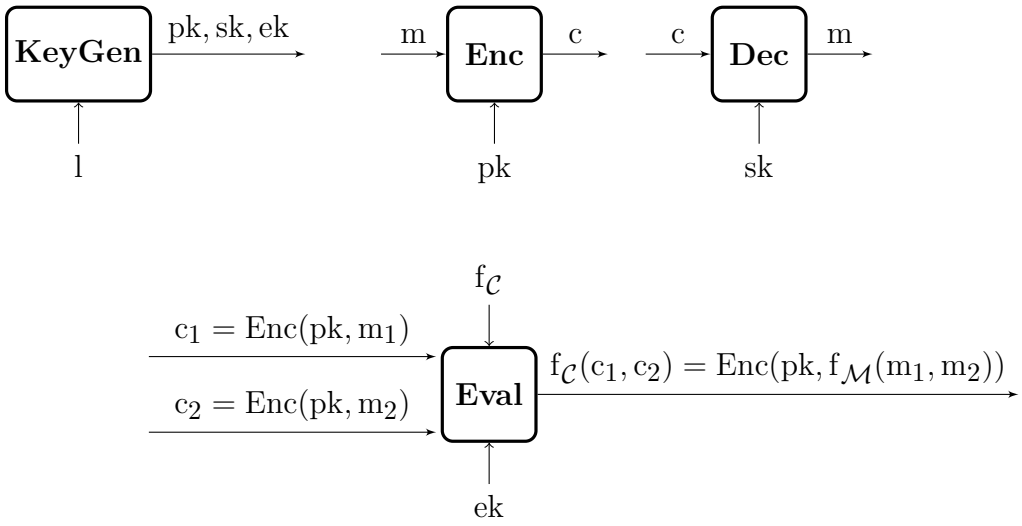

Figura 4.3: Componentes de um sistema FHE

O sistema proposto em [75], por exemplo, é utilizado para construir um protocolo de computação segura por múltiplas partes que pode ser utilizado para rodar funções arbitrárias sobre conjuntos de documentos encriptados com múltiplas chaves, impondo a maior parte da carga computacional sobre o servidor na nuvem. O resultado da função é um valor encriptado, que pode ser decifrado no dispositivo do usuário. Mesmo utilizando extensivamente o processamento na nuvem, esse protocolo ainda exige um grande número de interações com o usuário, o que o torna pouco prático.

Ainda não existem, no entanto, esquemas plenamente homomórficos que sejam suficientemente práticos e eficientes para que tenham aplicação geral, e possam ser adotados pelo consumidor comum da nuvem. Todavia, a pesquisa nesse campo tem produzido alternativas bastante eficientes com esquemas parcialmente homomórficos. Esses sistemas apresentam homomorfismos em operações básicas, como adição ou multiplicação, ou em relações de outra natureza, tal como a ordem dos elementos.

A combinação de esquemas assim pode prover todas as necessidades básicas de computação para um determinado domínio de conhecimento. No caso do experimento conduzido nessa pesquisa, considerou-se que a aplicação deveria ser capaz de realizar, de maneira eficiente, o cálculo de medidas de tendência central (média, mediana, mediatriz) e a seleção de registros que contivessem entradas contidas em determinados intervalos numéricos. A partir desse entendimento, foram selecionados criptossistemas que permitissem pelo menos a execução de adições e a comparação de ordem entre dados numéricos na nuvem. Daí a busca por um esquema aditivamente homomórfico e por um esquema com preservação de ordem.

4.2.3 Esquemas aditivamente homomórficos

Um esquema criptográfico é aditivamente homomórfico – *Additively Homomorphic Encryption* (AHE) – se existir uma operação $\oplus$ sobre $\mathcal{C}$, a imagem da função de cifra Enc(), de forma que, para quaisquer duas mensagens, a cifra de sua soma for igual à execução de $\oplus$ sobre suas respectivas cifras. Assim:

$$\forall \mathrm{m}_1, \mathrm{m}_2 \in \mathcal{M}, \tag{4.7}$$

$$\mathrm{c}_1 = \mathrm{Enc}(\mathrm{m}_1), \mathrm{c}_2 = \mathrm{Enc}(\mathrm{m}_2) \Rightarrow \mathrm{Enc}(\mathrm{m}_1 + \mathrm{m}_2) \equiv \mathrm{c}_1 \oplus \mathrm{c}_2 \tag{4.8}$$

Como exemplos de sistemas criptográficos com essa propriedade temos o esquema de Benaloh, que é baseado na primitiva *Dense Probabilistic Encryption* (DPE) e, portanto, exige cuidados especiais na geração da chaves [44], e o sistema de Melchor *et al* [82], que é baseado num encadeamento de cifras de reticulado. Neste último, o número de adições consistentes é dependente da extensão da cadeia.

Outro esquema aditivamente homomórfico que recebe bastante atenção na literatura e conta com diversas implementações foi proposto por Pascal Paillier em 1999 [89]. Esse criptossistema apresenta as seguintes propriedades:

- **Criptografia assimétrica** - é possível realizar computações homomórficas sobre os dados tendo posse apenas da chave pública.

- **Homomorfismo aditivo** - o resíduo quadrático da multiplicação de duas cifras é igual à cifra do resíduo da soma de suas respectivas mensagens. Isto é,

$$\text{Enc}(m_1).\text{Enc}(m_2) \mod n^2 = \text{Enc}(m_1 + m_2 \mod n) \tag{4.9}$$

- **Homomorfismo multiplicativo** - uma cifra elevada a uma mensagem é igual à cifra da multiplicação das duas mensagens. Isto é,

$$\text{Enc}(m_1)^{m_2} = \text{Enc}(m_1.m_2 \mod n) \tag{4.10}$$

O sistema é definido pelos três algoritmos:

Paillier.KeyGen o algoritmo de geração de chaves seleciona dois primos grandes p, q; computa seu produto $n = pq$; seleciona uniformemente um inteiro coprimo com n, i.e. $g \xleftarrow{u} \mathbb{Z}_n$; computa $\lambda = \text{mmc}(p-1, q-1)$; e, por fim computa $\mu = \frac{1}{L(g^\lambda \mod n^2)}$, onde $L(u) = \frac{u-1}{n}$;

A chave pública tem a forma $\langle n, g \rangle$, e a privada $\langle \mu, \lambda \rangle$;

Paillier.Enc dadas a chave pública $\langle n, g \rangle$ e uma mensagem $m \in \mathbb{Z}_n$, o algoritmo de cifragem consiste em selecionar uniformemente $r \xleftarrow{u} \{1..n-1\}$ e computar a cifra $c = g^m r^n \mod n^2$

Paillier.Dec o algoritmo de decifragem, por sua vez, recebe a chave privada $\langle \mu, \lambda \rangle$ e uma cifra c e computa a mensagem correspondente da seguinte forma: $m = L(c^\lambda \mod n^2).\mu \mod n$.

Esse sistema foi escolhido por sua simplicidade: a implementação depende apenas de operações matemáticas presentes em qualquer biblioteca criptográfica, tais como exponenciação modular e teste de primariedade.

A segurança do sistema de Paillier se baseia no Problema Decisional da Residuosidade Composta (DCRP, na sigla em inglês). O problema DCRP trata da dificuldade de decidir, dados um composto $n = pq$, onde p e q são primos grandes, e um inteiro g coprimo com n (i.e. $g \in \mathbb{Z}_n$), se g é um n-ésimo resíduo módulo n^2. Em outras palavras, o problema consiste em encontrar $y \in \mathbb{Z}^*_{n^2}$ tal que $g \equiv y^n \mod n^2$. No trabalho em que introduziu esse sistema, Paillier definiu o problema DCRP e demonstrou sua equivalência (em termos de custo computacional) com o problema da fatoração de n, que dá base a outros sistemas, como o RSA.

No sistema de Paillier, a chave de avaliação da função homomórfica é elemento da chave pública: o servidor da nuvem só precisa conhecer o composto n, para executar as

multiplicações modulares. O custo das operações modulares pode ser uma preocupação nos casos em que houver a necessidade de realizar a soma de milhões de valores numa base grande. O trabalho de Nikolaenko *et al* mostra a construção de um protocolo baseado *garbled circuits*, que utiliza o sistema de Paillier para executar uma regularização de Tikhonov, ou regularização em cristas, um método de regressão linear, sobre milhões de registros cifrados [87].

No protótipo construído nesse trabalho nenhuma técnica especial precisou ser empregada, pois as execuções do homomorfismo do sistema de Paillier são internas ao prontuário do paciente: cada paciente tem sua chave, e, portanto, a soma só é consistente entre os registros de um prontuário. Operações estatísticas sobre toda a base exigem, portanto, um algoritmo que realize a consolidação interna de cada paciente como passo intermediário.

4.2.4 Esquemas com preservação de ordem

Esquemas criptográficos com preservação de ordem – *Order-preserving Encryption* (OPE), são aqueles em que as cifras mantêm a mesma relação de ordem que as respectivas mensagens. Isto é:

$$\forall \mathrm{m}_1, \mathrm{m}_2 \in \mathcal{M}, \tag{4.11}$$

$$\mathrm{m}_1 \prec \mathrm{m}_2 \Rightarrow \mathrm{Enc}(\mathrm{m}_1) \prec \mathrm{Enc}(\mathrm{m}_2) \tag{4.12}$$

Alguns exemplos de sistemas e protocolos de cifra com preservação de ordem foram propostos em [94] e [79]. O sistema de Boldyreva *et al* utiliza as propriedades da função de distribuição de probabilidade hipergeométrica para fazer o espalhamento das cifras num espaço maior que o domínio (o espaço de mensagens $\mathcal{M}$) [18]. É um sistema simétrico e determinístico: a mesma chave cifra e decifra os dados e, para uma chave fixa, cada número no espaço de mensagens é cifrado sempre para o mesmo número no espaço de texto cifrados. Com o objetivo de facilitar a leitura, esse sistema é sempre referenciado como esquema de OPE de Boldyreva neste trabalho, embora ela não seja a única autora do trabalho.

A distribuição hipergeométrica é aquela que modela problemas em que se deseja determinar a probabilidade condicional iterada de um evento. O exemplo comumente utilizado é a probabilidade de retirar um número x de bolas brancas de uma urna com B bolas brancas e P bolas pretas após um número n de retiradas. Essa função de distribuição de probabilidade é geralmente expressa na seguinte forma:

$$\mathrm{P}[\mathrm{X} = \mathrm{k} | \mathrm{B}, \mathrm{P}, \mathrm{n}] = \frac{\binom{\mathrm{B}}{\mathrm{x}}\binom{\mathrm{P}}{\mathrm{n-x}}}{\binom{\mathrm{B+P}}{\mathrm{n}}} \tag{4.13}$$

O sistema de Boldyreva utiliza essa característica para projetar intervalos do espaço de mensagens sobre o espaço de cifras, reduzindo iterativamente os intervalos de interesse. Na última iteração, quando o intervalo de entrada não puder mais ser reduzido, o sistema utiliza a distribuição uniforme sobre o intervalo de saída correspondente. O funcionamento pode ser descrito pelos seguintes algoritmos:

Boldyreva.KeyGen um algoritmo com acesso a uma fonte de aleatoriedade, que reúna um dado número de bits com forte entropia. A implementação utilizada neste trabalho usa o mesmo gerador selecionado para a cifra AES e produz chaves de 256 bits de extensão;

Boldyreva.Enc dados a chave, um espaço de entrada $\mathcal{M}$, um espaço de saída $\mathcal{C}$ e a mensagem m $\in \mathcal{M}$, reduz iterativamente os espaços de entrada e saída, até que haja apenas o valor m no espaço de entrada (i.e. $\mathcal{M} = \{\text{m}\}$). Como último passo, seleciona um elemento do espaço de saída conforme um distribuição contínua 'enviesada' por uma semente gerada pelo algoritmo OPE.TapeGen.

Boldyreva.Dec executa exatamente o mesmo algoritmo de ciframento, com exceção do último passo, em que, em lugar do resultado no contradomínio, retorna o primeiro valor no domínio.

É a função TapeGen, descrita no algoritmo 4.2 que torna o sistema determinístico. Ela também dá base à noção de segurança definida para o sistema. É uma função pseudoaleatória de saída variável (extensão igual ou superior ao parâmetro de segurança l), que resulta da composição de uma função pseudoaleatória com tamanho de entrada flexível (HMAC-SHA256) e uma cifra de bloco (AES-256-CBC).

A noção de segurança do sistema se baseia na ideia de que a função pseudoaleatória com entropia alta distribui os intervalos mapeados de forma indistinguível: isto é, um atacante não pode decidir, em tempo polinomial, se um dado intervalo foi selecionado aleatoriamente ou se resulta do processo de decisão orientado pela função pseudoaleatória TapeGen. Boldyreva *et al* introduzem o conceito de segurança IND-OCPA, que significa que um atacante não pode aprender nenhuma informação ao observar as cifras produzidas pelo sistema além da ordem das mensagens cifradas. É uma proposição inatingível por esse sistema, pois o algoritmo também é determinístico: o que faz com que o atacante aprenda não somente a ordem, como também a existência de duplicatas [19].

É evidente que a vantagem do atacante nunca será desprezível (como em sistemas CPA seguros, ou com indistinguibilidade semântica). Mas esse tipo de sistema não pode ser analisado sob os mesmos parâmetros usadas para analisar sistemas com indistinguibilidade do texto cifrado – uma das características mínimas de segurança exigidas de cifras de uso

Algoritmo 4.2. OPE (Boldyreva)

Require: l parâmetro de segurança

function ENC(Key, $\mathcal{M}, \mathcal{C}$, m)
  M ← $|\mathcal{M}|$
  N ← $|\mathcal{C}|$
  d ← min($\mathcal{M}$) – 1
  r ← min($\mathcal{C}$) – 1
  y ← r + ⌈fracN2⌉

  if M = 1 **then**
    coins ← TapeGen(Key, l, $\mathcal{M}, \mathcal{C}$, m)
    c $\xleftarrow{\text{u(coins)}}$ $\mathcal{C}$
    return c
  end if
  coins ← TapeGen(Key, l, $\mathcal{M}, \mathcal{C}$, y)
  x ← d + HG(M, N, y – r, coins)
  if m ≤ x **then**
    $\mathcal{D}$ ← {d + 1, ..., x}
    $\mathcal{C}$ ← {r + 1, ..., y}
  else
    $\mathcal{D}$ ← {x + 1, ..., d + M}
    $\mathcal{C}$ ← {y + 1, ..., r + N}
  end if

  return Enc(Key, $\mathcal{M}, \mathcal{C}$, m)
end function

function TAPEGEN(Key, l, $\mathcal{M}, \mathcal{C}$, coin)
  while |vol_prf| < l **do**
    vil_prf ← HMAC(Key, coin)
    coin ← AES.Enc(Key, vil_prf)
    vol_prf ← vol_prf||coin
  end while
  return vol_prf
end function

function DEC(Key, $\mathcal{M}, \mathcal{C}$, c)
  M ← $|\mathcal{M}|$
  N ← $|\mathcal{C}|$
  d ← min($\mathcal{M}$) – 1
  r ← min($\mathcal{C}$) – 1
  y ← r + ⌈fracN2⌉

  if M = 1 **then**
    coins ← TapeGen(Key, l, $\mathcal{M}, \mathcal{C}$, m)
    c′ $\xleftarrow{\text{u(coins)}}$ $\mathcal{C}$
    if c′ = c **then**
      return min($\mathcal{M}$)
    end if
    return ⊥
  end if
  coins ← TapeGen(Key, l, $\mathcal{M}, \mathcal{C}$, y)
  x ← d + HG(M, N, y – r, coins)
  if m ≤ x **then**
    $\mathcal{D}$ ← {d + 1, ..., x}
    $\mathcal{C}$ ← {r + 1, ..., y}
  else
    $\mathcal{D}$ ← {x + 1, ..., d + M}
    $\mathcal{C}$ ← {y + 1, ..., r + N}
  end if

  return OPE.Dec(Key, $\mathcal{M}, \mathcal{C}$, c)
end function

geral. A questão é que o projeto básico do sistema é feito para que alguma informação sobre os valores cifrados seja exposta: a sua ordem.

A proporção de bits revelados é uma das considerações de projeto de cifras com essa propriedade. Em geral, essa proporção é função inversa do custo de ciframento do dado. O trabalho de Popa *et al* mostra que, a depender da primitiva que dá base ao sistema, da dispersão dos dados no domínio e do número de amostras disponíveis para comparação (cifras geradas com a mesma chave), um atacante pode descobrir a ordem de magnitude do dados cifrados com o sistema de Boldyreva, e revelar uma proporção considerável dos bits do texto plano. Naqueles sistemas baseados em primitivas determinísticas, essa proporção chega a ser superior a 50% dos bits [94].

O mesmo trabalho propõe o protocolo stOPE, que apresenta um resultado muito mais forte. O protocolo resulta na criação de uma árvore binária de busca, com nós de tamanho fixo, produzidos por meio de um sistema simétrico não determinístico. Cada execução do algoritmo de ciframento exige pelo menos uma busca sobre essa árvore – um custo de ordem logarítmica sobre o número de valores cifrados anteriormente. Para cada passo da busca (cada nível da árvore, começando da raiz), uma execução do algoritmo de deciframento é executada, para saber o valor do nó corrente e decidir se o novo nó deve ser posto à esquerda ou à direita do corrente. Em alguns casos, a inserção do novo nó causará um rebalanceamento da árvore. A cifra simétrica não tem preservação de ordem e não revela qualquer informação acerca do valor cifrado. A ordem está na construção da árvore. O servidor na nuvem nunca aprende qualquer informação além da ordem de inserção dos nós. É um sistema muito mais seguro. Em contrapartida é extremamente custoso.

A escolha pelo sistema de Boldyreva se deu pelo fato de que é um sistema muito mais eficiente e o nível de segurança foi considerado é adequado para o campo de aplicação específico analisado no decurso desta pesquisa. Os dados não têm um espalhamento muito grande. E são cifrados com chaves diferentes para cada prontuário, o que diminui drasticamente a quantidade de valores cifrados com a mesma chave e, por conseguinte, a capacidade do atacante de criptoanalisar os resultados.

4.3 Criptografia determinística

Diz-se que um esquema criptográfico é determinístico se, para qualquer mensagem m no espaço $\mathcal{M}$, existe apenas uma cifra c no espaço de texto cifrado $\mathcal{C}$ que seja igual ao resultado da função de ciframento.

$$\forall \mathrm{m} \in \mathcal{M}, \exists ! \mathrm{c} \in \mathcal{C} | \mathrm{c} = \mathrm{Enc}(\mathrm{m}) \tag{4.14}$$

Uma maneira prática de construir um esquema determinístico é utilizar um vetor de inicialização fixo em construções com cifra de bloco, como os modos CBC e CTR. Esse tipo de cifra não atende ao requisito de indistinguibilidade semântica, pois um observador sempre será capaz de identificar valores duplicados, caso existam. Mais uma vez, é importante lembrar que esse relaxamento na condição de segurança é parte do projeto do sistema, e não representa uma falha. Como discutido anteriormente, no caso de sistemas OPE, o perigo reside no uso incorreto das funcionalidades providas por essas primitivas e não em sua construção, com as respectivas definições ou condições de segurança.

O Safe-Record usa o AES-256-CBC, com um *hash* da identificação da instalação (uma função de dados do provedor, versão, etc) como chave e um iv fixo, selecionado durante a implantação. São cifrados com esse sistema determinístico os identificadores de objetos de banco de dados (nomes de tabelas e *views*, nomes de campos), e outros dados fixos para todos os usuários.

Esse sistema realmente não substitui a cifra simétrica utilizada como mecanismo de confidencialidade dos dados privados. Ele é usado apenas como uma barreira extra de segurança, um complicador, que exige do atacante uma dedicação mais longa ou um poder maior de computação para ler as etiquetas, os sinalizadores, que ajudam a entender o contexto e possíveis significados do dado cifrado e, assim, criptoanalisá-lo. O objetivo é gerar um efeito de 'ofuscação' do sistema: uma técnica importante, que, indiretamente, eleva o nível da segurança dos dados [90, 56].

É claro que um atacante com acesso ao banco, ao código da aplicação e aos parâmetros de configuração do servidor poderia decifrar esses identificadores de maneira trivial. Mas veja que, caso do Safe-Record, isso já exige que o atacante tenha sucesso contra sistemas distintos: o atacante precisaria ter acesso tanto ao servidor da nuvem onde roda a aplicação, quanto ao provedor da nuvem que provê o serviço de banco de dados. Note ainda que se tratam de dados públicos. Os dados privados e sigilosos estarão sempre cifrados com AES-256-GCM, com as chaves individuais dos pacientes.

5 Safe-Record

O trabalho exposto neste Capítulo se refere ao exercício de análise realizado para projetar e construir um protótipo de S-RES como prova de conceito da aplicação dos esquemas e primitivas criptográficos apresentados no capítulo anterior. Esse protótipo foi intitulado Safe-Record: o nome surgiu da uma proposta de segurança para Registros Eletrônicos em Saúde, pois faz alusão direta a registros em saúde, sendo que a palavra 'safe' traz, ainda, acepções relacionadas à segurança e guarda dos dados.

O Capítulo 2 deste trabalho apresenta diversas vulnerabilidades de ambientes típicos de computação em nuvem, inclusive problemas ainda não resolvidos, tais como a *insider threat* – o risco de que o provedor acesse indevidamente os dados hospedados em sua nuvem. A solução recorrentemente apresentada na literatura é a encriptação dos dados no lado do cliente, sempre que houver requisitos estritos de custódia dos dados [70, 119, 113].

O projeto da solução apresentada nesse trabalho acrescenta a esta intuição inicial, a percepção de que, quaisquer que sejam as vantagens dos esquemas criptográficos e outros mecanismos de segurança propostos, sua introdução jamais pode dificultar ou até mesmo impedir o alcance dos requisitos de negócio que definem a operação cotidiana das aplicações reais, do domínio semântico da informação que se pretende proteger. Caso contrário, a proposta pode representar uma contribuição muito restrita para o desenvolvimento do conhecimento em segurança da informação e, consequentemente, para a segurança de aplicações reais.

> The most profound technologies are those that disappear. They weave themselves into the fabric of everyday life until they are indistinguishable from it [117].
>
> *As tecnologias mais profundas são aquelas que desaparecem. Elas vão se entrelaçando no tecido da vida cotidiana até que sejam indistinguíveis desta.* (tradução livre) – Mark Weiser.

A aplicação deve, por exemplo, apresentar uma funcionalidade para a realização de buscas, comparações e ordenações dos registros de maneira eficiente. Uma solução baseada no armazenamento dos registros de um paciente como um bloco encriptado, onde uma busca implica no *download* e decifragem de todos os registros, não atende a esse requisito básico e, consequentemente, não pode ser usada em uma aplicação real.

Após extensa revisão de literatura, não foi encontrada uma solução que atendesse a esses requisitos mínimos. A maior parte das soluções cifra o prontuário como um bloco,

com criptossistemas que não permitem computações sobre os dados cifrados na nuvem [50, 26, 14, 85, 65]. As soluções que incorporam sistemas criptográficos homomórficos, e, portanto, permitem alguma computação segura na nuvem, também não são realmente aplicáveis à indústria de atenção à saúde [35, 116, 20, 95]. Principalmente porque não levaram em conta as propriedades dos modelos semânticas consensuais do domínio dos dados que estão sendo criptografados. Essas soluções não permitem, por exemplo, as buscas e operações típicas de um RES modelado de acordo com os padrões OpenEHR.

Este trabalho apresenta um contribuição para a área, pois adota uma abordagem de segurança que parte da análise da natureza dos dados para, a partir da definição das propriedades da massa de dados e das operações que são necessariamente executadas sobre eles, selecionar os criptossistemas mais adequados.

Essa é a motivação do esforço de pesquisa apresentado no Capítulo 3, que examina características importantes do OpenEHR. O OpenEHR é um conjunto de padrões para modelagem de dados clínicos que tem ampla e crescente aceitação na academia e na indústria. De forma sucinta, a proposição resultante daquela fase do trabalho é que qualquer proposta de segurança para um sistema RES construído sob os padrões OpenEHR deve, no mínimo, permitir a manipulação de dados numéricos na nuvem, sem perda da confidencialidade e integridade da informação clínica.

Não se pode ignorar, no entanto, a necessidade de um controle de acesso robusto. Daí a seleção de sistemas e primitivas criptográficas apresentada no quarto capítulo, o quais foram agrupados de acordo com esses requisitos primordiais: controle de acesso aos dados e manipulação de numéricos encriptados na nuvem.

5.1 Projeto

Diante dessas considerações, o projeto do Safe-Record, a prova de conceito deste trabalho, resultou na construção de três artefatos de software:

- CloudEHR: uma aplicação *web* para gestão de RES, projetada para ser implantada em serviços de PaaS;
- MobileID: um aplicativo para dispositivos móveis, usado no protocolo de autenticação por dois fatores;
- BrowserTrust: uma extensão para *browsers* comerciais comuns, utilizada como elemento confiável na verificação do código rodando no cliente.

A Figura 5.1 dá uma idéia da organização e comunicação desses artefatos. Os detalhes de projeto e de sua construção, vem a seguir, bem como a explanação detalhada de sua participação na segurança da solução.

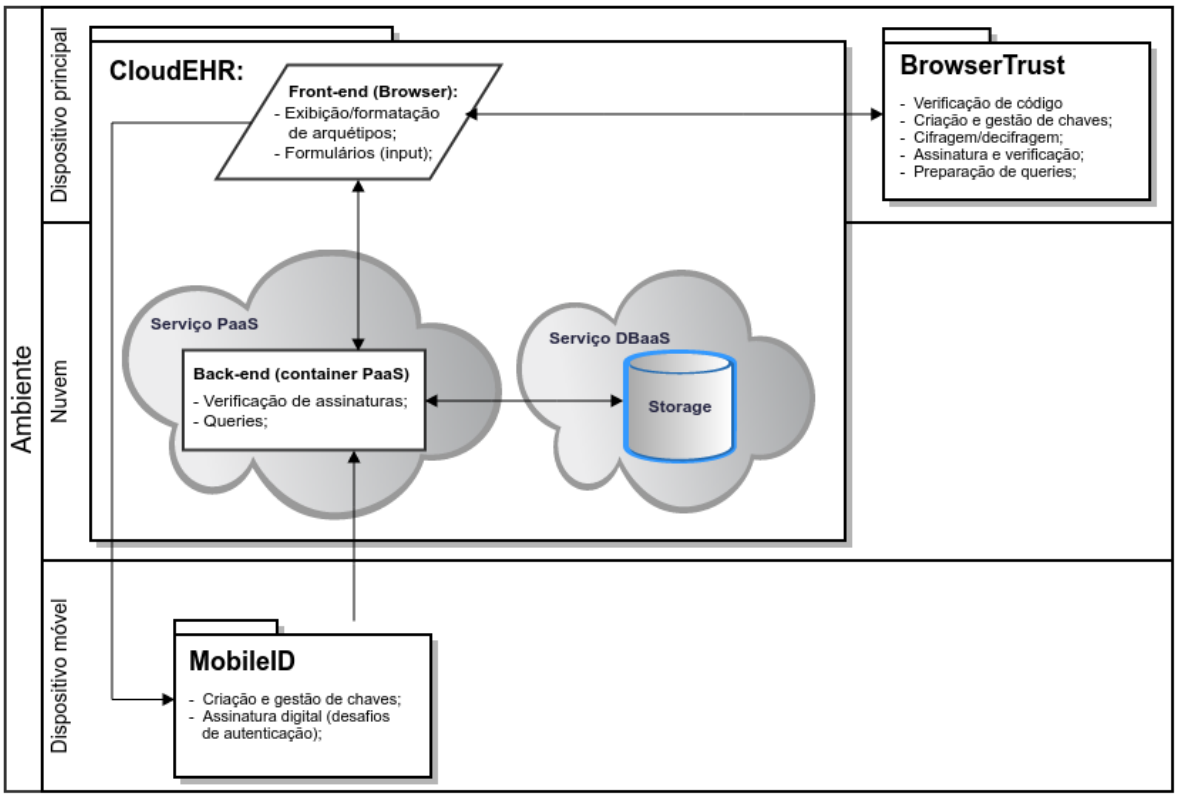

Figura 5.1: Arquitetura do Safe-Record

O projeto da solução pode ser expresso em duas fases, ou dois grandes grupos de funcionalidades: um para controle de acesso e outro para manipulação dos RES na nuvem. O requisito primário considerado foi que o controle de acesso não deveria impor qualquer condição à gestão dos registros em saúde. Daí a necessidade de propor um ferramental de controle de acesso que fosse flexível o bastante para atender a qualquer especificidade da aplicação, sem perda de segurança.

Ademais, foi feita a opção por uma política de controle de acesso baseada em papéis, de acordo com o que é proposto em [122]. O trabalho de Xiao *et al* introduz o conceito de "Princípio de acesso": cada elemento do RES deve ser associado a uma lista de usuários que podem acessá-lo. Também introduz o "Princípio de controle": cada elemento deve ser associado a um responsável pela informação, que tem a capacidade de alterar a lista de controle de acesso. O Safe-Record atinge esses objetivos com o uso de primitivas criptográficas: a lista de permissões de acesso se materializa com o protocolo de compartilhamento das chaves descrito na Seção 5.3.3, usado para dar acesso às chave criptográficas utilizadas na cifra dos registros clínicos. O responsável pelo RES é o próprio paciente, que tem a guarda de suas chaves e pode ver e alterar a lista a qualquer momento.

A intenção era utilizar uma técnica de particionamento virtual da aplicação de forma

que, mesmo que uma vulnerabilidade em uma funcionalidade específica fosse explorada ou mesmo que a identidade de um usuário específico fosse roubada, apenas um subconjunto restrito dos dados pudesse ser afetado [47]. O nível de particionamento desejado foi atingido, neste trabalho, com o uso de chaves criptográficas diferentes para cada paciente.

Por isso, a aplicação servida pela nuvem foi projetada sob um paradigma cliente-servidor. O *front-end* servido pela nuvem, e que roda num *browser* comum, não é apenas uma interface, mas contém todos os elementos de uma aplicação no padrão arquitetural MVC (Model-View-Controller). Na camada de modelo temos o acesso aos dados, bem como suas respectivas regras de negócio: incluso a cifragem e decifragem de todos os dados trocados com a nuvem e conversão dos modelos relacionais persistidos na nuvem para o modelo de objeto de arquétipo do OpenEHR. Na camada de controle, temos o controle de fluxo e de mensageiria entre os diversos módulos do cliente. Na camada de visão, por fim, as classes responsáveis por renderizar os objetos de interface bem como por reagir à interação do usuário.

É importante notar que, apesar da criptografia no lado do cliente, a intenção desse trabalho é produzir uma proposta que permita aproveitar o poder computacional da nuvem para realizar algumas operações criptográficas, especialmente a realização de buscas sobre os dados encriptados. Por isso, o lado do servidor também foi escrito sob o padrão MVC e conta com controles de fluxo e de acesso. Isso inclui a execução operações complexas, tais como o protocolo SRP, a verificação de assinaturas no criptossistema RSA e ainda a execução de somas sobre dados cifrados com o sistema de Paillier.

O trabalho de Gonçalves *et al* [50], do qual participaram discentes e pesquisadores deste Departamento de Engenharia Elétrica, foi examinado minuciosamente e nele se fundam os conceitos básicos, tais como a definição dos papéis, a mecânica do processo de autorização e o procedimento de "*break-the-glass*" (autorização especial, em casos de emergência), do protocolo de controle de acesso apresentado a seguir.

O principal conceito naquele trabalho é o contrato, que representa uma relação de confiança ou uma autorização estabelecida entre dois usuários e se concretiza com o compartilhamento das chaves criptográficas necessárias para o acesso aos dados de um usuário. Médicos, por exemplo, detêm uma cópia das chaves utilizadas na encriptação do registro de seus respectivos pacientes. Um esquema assimétrico foi utilizado, de forma que o paciente pudesse encriptar suas chaves com a chave pública do médico, garantindo que somente aquele médico autorizado pudesse decifrar as chaves e, então, acessar o RES.

Embora utilize os mesmos conceitos básicos, o de controle de acesso do Safe-Record difere da proposta de Gonçalves *et al* em questões extremamente importantes:

- Em Gonçalves, a senha do usuário é utilizada diretamente no sistema PBE utilizado para encapsular as chaves criptográficas do usuário. No Safe-Record a senha

é pré-processada com uso do algoritmo PBKDF2, expandindo significativamente a entropia e, consequentemente, a segurança do sistema resultante.

- Em Gonçalves, o desafio de autenticação armazenado na nuvem é um *hash* da senha concatenada a sequência de *salting*, que é frágil diante de um ataque do tipo 'força bruta', o qual pode ser utilizado para recuperar a senha do usuário e derrotar completamente a segurança do componente de controle de acesso. No Safe-Record, a prova de conhecimento da senha do usuário é executada por meio do protocolo SRP, no qual as informações persistida no servidor ou trocadas na execução do protocolo são insuficientes para reconstituição da senha;

- Em Gonçalves, a prova de autoria do contrato é um HMAC (código de autenticação de mensagem baseado em *hash*), que, por se basear em uma função de *hash* de propósito geral, sofre da mesma vulnerabilidade que o desafio de autenticação. No Safe-Record, a prova de autoria é uma assinatura digital RSA. Além de conferir maior segurança à solução, essa escolha também traz mais flexibilidade - dado o maior número de funcionalidades associadas à ICP que pode ser derivada do criptossistema RSA, bem como às características da primitiva de assinatura digital em si;

- Em Gonçalves, o par de chaves RSA utilizado para identificar o dispositivo móvel que responde aos desafios de autenticação é gerado na nuvem e entregue ao dispositivo na forma de um código QR visível na tela. As chaves podem ser capturadas pelo provedor e usadas em ataques para roubo de identidade do usuário. Além disso, uma vez em tela, o código pode ser gravado e decodificado posteriormente por qualquer atacante, em qualquer dispositivo, comprometendo seriamente a segurança da aplicação. No Safe-Record, todos os processos de geração e manipulação de chaves se dão no dispositivo do usuário. As chaves do dispositivo móvel, em especial, são geradas no próprio dispositivo;

- Em Gonçalves, a aplicação rodando na nuvem envia notificações diretamente ao dispositivo móvel durante o processo de autenticação. Nos sistemas operacionais mais utilizados (e.g. Android, iOS, Tzen), notificações de aplicativos devem passar necessariamente pelo servidor de notificações oficial da plataforma. Ao utilizar esse serviço, a privacidade do usuário é comprometida, pois o mantenedor da plataforma será notificado sempre que o usuário acessar essa aplicação relacionada à saúde. No Safe-Record, o desafio é exibido no monitor, lido pelo dispositivo móvel e a resposta é enviada diretamente à aplicação na nuvem. Dessa forma, apenas agentes necessariamente envolvidos no contexto da aplicação estarão cientes da existência dessa comunicação.

5.1.1 Atores e requisitos funcionais

Os requisitos são enunciados a seguir, classificados de acordo com os papéis a que estão associados. O requisito é descrito na forma de uma ação ou funcionalidade disponível para aquele tipo de usuário:

- **Usuário não autenticado:** qualquer usuário, acessando a aplicação em qualquer de suas interfaces públicas (i.e. aplicativo *web*, lojas de aplicativos, etc.):
  1. Instalar o aplicativo no dispositivo móvel;
  2. Instalar a extensão no *browser*;
  3. Acessar o aplicativo *web*;
  4. Iniciar uma instância do protocolo de autenticação (no *browser*);
  5. Receber e responder a desafios de autenticação (no dispositivo móvel);
- **Paciente:** é um usuário autenticado, associado a um perfil de paciente p_i, que mantém registros de saúde privados e pode compartilhá-los com clínicos de sua escolha. Seus requisitos são:
  1. Ler todos os seus registros clínicos;
  2. Ler e alterar seus registros demográficos;
  3. Gerir seus contratos: listar, suspender, restabelecer ou revogar as autorizações de acesso a seu registro.
- **Clínico:** um profissional da área de saúde, identificado como d_i, que é autorizado a gerir registros de pacientes específicos.
  1. Credenciar um paciente;
  2. Requisitar autorizações de acesso a pacientes credenciados por outros clínicos;
  3. Ler, inserir e alterar registros clínicos e demográficos, nos prontuários a que está autorizado.
- **Gestor:** um profissional atuando como responsável ou supervisor dos demais clínicos. No Brasil, a legislação apresenta os papéis de Diretor Técnico, Diretor Clínico e Chefe de Serviço ou Platão - definidos nas Resoluções 1.342/1991 e 2.056/2013, do Conselho Federal de Medicina (CFM). Em resumo, trata-se de um médico especialista legalmente habilitado, que realiza funções de supervisão técnica ou representação administrativa e trabalhista, e representa o elo entre a instituição prestadora de serviços de atenção à saúde e os médicos que nela atuam.

A maior parte dos países também exige que as instituições realizem revisões periódicas de seus prontuários. No Brasil, essa prática é imposta pela Resolução 1.638/2002 do CFM, que determina a criação de uma comissão revisora em todas as IAS. Ademais, os sistemas de RES têm um requisito conhecido na literatura como *break-the-glass procedure*, que consiste na existência de um método de acesso emergencial ao registro [7]. Essa funcionalidade se tornou obrigatória no Brasil com a Resolução CFM 1.821/2007, que instituiu o prontuário médico eletrônico.

A figura do clínico gestor é importante para desempenhar essas funções especiais de controle de acesso. Portanto, os requisitos associados ao gestor são:

1. Gerir os dados demográficos de sua instituição de atenção a saúde (IAS);
2. Credenciar clínicos ou associar clínicos à sua IAS;
3. Acessar os dados criados pelos clínicos no âmbito de sua IAS (revisão de prontuário);
4. Conceder acessos emergenciais (*break-the-glass procedure*).

5.1.2 Modelo adversarial

A fim de orientar o projeto e a construção do Safe-Record, também foram elicitadas as principais ameaças, de acordo com o modelo adversarial considerado.

Modelo adversarial típico:
Representa o ambiente onde o sistema está exposto à atividade regular dos usuários e à atividade maliciosa de agentes externos. Onde, atividade maliciosa é qualquer tentativa do agente externo de ganhar acesso ou alterar dados privados na aplicação;

Modelo com o provedor semi-honesto, ou 'honesto, mas curioso':
Cenário em que o provedor da nuvem, ou um agente interno ao provedor, também é um atacante potencial e é capaz de realizar computações arbitrárias sobre os dados na nuvem, além das análises e inferências estatísticas que normalmente já faria, utilizando informações de tráfego (endereços IP dos requisitantes; tamanho e cabeçalhos dos pacotes de requisição; etc.), consumo de recursos, e demais informações que são necessariamente expostas ao provedor;

Modelo com o provedor malicioso
Quando o provedor, ou um agente interno, efetivamente altera dados armazenados ou em processamento, ou ainda o conteúdo dos pacotes trocados entre usuários finais e a aplicação na nuvem.

O modelo adversarial típico exige providências suficientemente abordadas na literatura: um protocolo robusto de controle de acesso, com processos de autenticação e autorização baseados em primitivas criptográficas fortes; um controle de sessão imune contra ataques característicos do protocolo HTTP e das interfaces de aplicações *web*; prevenção contra ataques relacionados ao *input*, tais como injeção de código; XSS; CSRF; *phishing*; entre outros. Todos esses aspectos foram considerados. No entanto, uma vez que o foco deste trabalho é estudar medidas de proteção no cenário do provedor semi-honesto, a discussão que segue dará mais atenção a elementos mais relevantes nesse cenários.

Para garantir que o sistema permaneça seguro no modelo 'honesto, mas curioso', é necessário negar ao provedor a capacidade de fazer uso de qualquer dado persistido, em processamento ou trafegando na nuvem. Além disso, para garantir a privacidade do usuário, é necessário negar ao provedor acesso a qualquer informação que possa levar a sua identificação. Todo o projeto do Safe-Record é baseado no princípio da criptografia no lado do cliente. Isto é, nenhum dado privado é enviado à nuvem sem ter sido previamente criptografado [119].

O provedor, inevitavelmente, terá em suas mãos informações de acesso dos usuários finais, tais como detalhes das placas de rede dos dispositivos, endereços de rede, caminho percorrido pelos pacotes, etc. Mas tudo isso também poderia ser obtido por um atacante observando pacotes no ISP do usuário, mesmo que a aplicação fosse servida no *data-center* da IAS. O foco deste trabalho está nas soluções para a confidencialidade dos dados e a privacidade do usuário final num ambiente de computação em nuvem. A criptografia no lado do cliente entra aqui, portanto, como proteção contra uma provável "curiosidade" do provedor, que tem acesso privilegiado aos componentes da infraestrutura dado suporte à execução da aplicação.

Para melhorar o nível de segurança da aplicação quando considerado o modelo adversarial do provedor malicioso, foi introduzido um elemento confiável: BrowserTrust, uma extensão que deve ser instalada no *browser* do usuário e permite que o código servido pelo provedor seja verificado. Assim, qualquer ataque envolvendo alteração no código que roda no cliente é facilmente identificado. Como todos os dados na nuvem são cifrados, um ataque envolvendo a alteração de um bit sequer torna a informação inutilizável, o que também é facilmente detectado no lado do cliente.

Este trabalho não pretende, no entanto, cobrir completamente o modelo do provedor malicioso, e, por isso, não se estende na procura de soluções para problemas como a remoção permanente de dados no provedor, a negação ou instabilidade no nível de serviços, entre outros. Também não são cobertos ataques envolvendo código malicioso rodando previamente na máquina do usuário, que pode resultar na gravação do conteúdo digitado ou exibido em tela, e outras técnicas que inviabilizariam a segurança de qualquer

aplicação. A análise da segurança do Safe-Record parte do pressuposto de que o usuário tem uma máquina comum, com um *browser* comum, ambos em condições normais de funcionamento.

5.2 Controle de acesso

O controle de acesso do Safe-Record se baseia num protocolo de autenticação por dois fatores, bem como num sistema híbrido de cifra baseada em senha. A autenticação por dois fatores envolve, no primeiro passo, uma prova de conhecimento através do protocolo SRP-6a (RFC 5054 [111]) e, no segundo, uma prova de posse de um hardware específico, através de uma assinatura digital RSA-SSP (PKCS#1 v2.2) produzida em um dispositivo móvel credenciado. Ao fim do processo de autenticação, o usuário tem acesso a um pacote cifrado com as chaves criptográficas utilizadas na cifra dos dados clínicos.

O sistema PBE se baseia no algoritmo PBES2 do padrão PKCS#5 v2.0 (RFC 2898 [68]), e é construído pela composição do algoritmo de derivação de chaves PBKDF2 (PKCS#5 v.2,0), do encapsulamento de chave privada do padrão PKCS#8 v2.2 (RFC 5958 [114]) e do sistema de criptografia assimétrica RSA-OAEP (PKCS#1 v2.2). Esse sistema é utilizado no encapsulamento das chaves criptográficas e garante que, sem conhecimento da senha correta, o usuário não possa decifrar o pacote de chaves e, consequentemente, não possa decifra e ter acesso aos registros clínicos. O sistema RSA-SSP também é utilizado na produção de assinaturas digitais durante o procedimento de autorização.

5.2.1 Credenciamento

O registro de um usuário é um procedimento em que o aplicativo, primeiramente, gera R - uma sequência de bits recebidos de uma fonte de aleatoriedade. A implementação atual usa a biblioteca OpenSSL, que gera, na verdade, uma sequência pseudoaleatória, que, por simplicidade, doravante será referenciada como sequência aleatória. Essa sequência é utilizada para elevar a entropia do desafio de identificação do usuário. A sequência u_i.R e a senha u_i.S são usados na execução do algorítimo de registro, segundo o a definição do protocolo SRP na RFC 5054 [111], produzindo um *string* de verificação de senha, conforme descrito no capítulo 4. Esse sequência, referenciada doravante como u_i.SRP.v, será utilizada, juntamente com a identificação do usuário u_i.id e a sequência de *salting* u_i.R no processo de autenticação, descrito mais adiante.

O aplicativo também gera um par de chaves RSA, que será usado para controle de acesso às chaves criptográficas do usuário. Então, a sequência u_i.R e a senha u_i.S também são usados como entradas para uma implementação do algoritmo PBKDF2, que gera uma chave temporária pbekey, que será utilizada para encriptar a chave privada RSA u_i.RSA.sk

no formato PEM (*Privacy-enhanced Electronic Mail*), de acordo com o padrão PKCS#8. A chave pbekey nunca é gravada, em qualquer que seja o dispositivo. Ela deve ser gerada novamente, na presença da senha u_i.S e da sequência u_i.R, cada vez que o usuário for autenticado no sistema.

Também são geradas as chaves que serão usadas na cifra das instâncias dos arquétipos OpenEHR, sejam elas relacionadas a registros clínicos ou demográficos. Uma chave de 256 bits para o sistema AES, usado para cifrar todas as entradas; uma chave de 256 bits para o sistema OPE, usado na cifra de entradas numéricas; e um par de chaves para o sistema AHE de Paillier, que serão usadas para cifra e adições sobre dados numéricos. Note que os médicos também terão registros privados, pois administram seus dados demográficos. O gestores, administram, ainda, os dados de sua instituição.

O cliente então envia para a nuvem a identificação do usuário, a sequência de *salting*, a sequência de verificação da senha, a chave pública e o encapsulamento da chave privada. Ao persistir esses dados na nuvem o usuário é levado a registrar também o dispositivo móvel que será utilizado no processo de autenticação. Para isso, recebe um *token*, na forma de um código QR Code, que é exibido na tela. O usuário deve usar o aplicativo MobileID, rodando no dispositivo móvel para ler o código na tela. O aplicativo lê e decodifica o QR, produz uma assinatura digital do *token* usando o algoritmo RSA-PSS. Esses processo pode ser visto, de maneira sintética, no Algoritmo 5.1.

Algoritmo 5.1. Credenciamento de usuário

Require: médico ou gestor autenticado d_j

procedure BROWSER.REGISTER(u_i.id, u_i.S)
  u_i.R $\xleftarrow{u}$ $\{0,1\}^*$ ▷ tomado de uma fonte de aleatoriedade
  u_i.SRP.v ← SRP.Setup(u_i.R, u_i.S)
  u_i.RSA ← RSA.KeyGen()
  u_i.keySet ← Browser.GenerateKeySet() ▷ Chaves para os sistemas de Paillier, Boldyreva e AES
  encapsuled ← RSA.Enc(keyset, u_i.RSA.pk)
  pbekey ← PBKDF2(u_i.S, u_i.R)
  u_i.pem ← PKCS#8.EncryptPrivateKeyToPem(u_i.RSA.sk, pbekey)
  Cloud.StoreUser(u_i.id, u_i.R, u_i.SRP.v, u_i.RSA.pk, u_i.pem)
  token ← Cloud.GenerateRegistrarionChallenge()
  Mobile.Register(u_i.id, token)
  Browser.CreateContract(d_j.id)
end procedure

procedure MOBILE.REGISTER(u_i.id, token)
  mob_i.RSA ← RSA.KeyGen()
  response ← RSA.Sign(token, mob_i.RSA.sk)
  Cloud.StoreDevice(u_i.id, mob_i.id, mob_i.RSA.pk, response)
end procedure

O último passo do processo é o estabelecimento de um contrato com a autoridade credenciadora. Isto é, no caso do paciente, será criado um contrato com médico que iniciou o credenciamento. No caso de um, médico, com o gestor que está realizando sua inclusão no sistema. Esse contrato representa uma relação de confiança implícita: isto é, indica que o usuário confia na autoridade credenciadora. No caso do paciente,

essa autorização implícita também se estende ao supervisor do médico credenciador. A mecânica do processo de autorização, são explicadas mais à frente.

Note que esse processo de derivar uma chave criptograficamente forte a partir da senha e, com ela, cifrar a chave que encapsula a chaves que, por fim, efetivamente cifram os dados é equivalente a um esquema PBE híbrido. O sistema final é resultado da composição de algoritmos padronizados, extensivamente testados e comprovadamente seguros. A implementação do Safe-Record, diferentemente de outros sistemas PBE, permite que o usuário altere sua senha de acesso a qualquer momento, sem perda de dados e sem a necessidade de descriptografar e recriptografá-los com a nova senha. Para isso, basta utilizar a nova senha para gerar uma nova sequência de verificação u_i.SRP.v, derivar uma chave temporária com a função PBKDF2 e cifrar novamente a chave RSA privada. As chaves encapsuladas com a chave RSA privada, bem como os dados cifrados com elas, permanecem inalterados.

O protocolo SRP não apenas protege o sistema contra ataques de dicionário contra a base de senhas, mas também impede ataques de *replay*, quando o atacante repete o pacote com o código de verificação enviado pelo cliente genuíno. Mesmo que o protocolo falhasse, não haveria grandes consequências para o sistema, pois o atacante receberia um pacote encriptado e, sem a senha, não seria capaz de avançar. Por outro lado, o sistema de autenticação baseado apenas em senha está sujeito a outras fragilidades. Especialmente por que os usuários costumam utilizar a mesma senha em diversos serviços e uma falha em um sistema externo poderia acabar revelando a senha usada no contexto do Safe-Record. Para dirimir os riscos derivados dessas fragilidades, um segundo fator de autenticação foi introduzido.

Por isso, outro ponto importante desse processo é o credenciamento do dispositivo móvel, uma vez que o usuário não será capaz de acessar a aplicação sem um dispositivo habilitado a responder aos desafios de autenticação. Pelo mesmo motivo, o usuário deve ter sempre no mínimo um dispositivo credenciado. Se perder o único dispositivo habilitado, deve recorrer à autoridade credenciadora (o clínico ou gestor que o credenciou) a fim de inserir um novo dispositivo em sua conta. O par de chaves RSA gerado no dispositivo e usado na assinatura dos *tokens* de autenticação é gravado no formato PEM encriptado do padrão PKCS#8, na área de armazenamento privado do aplicativo, reservada pelo sistema operacional.

Todos os usuário têm acesso a uma funcionalidade que lista os dispositivos credenciados. O usuário pode, a qualquer momento, credenciar ou descredenciar um dispositivo. Apenas duas condições são impostas: 1) cada usuário deve ter no mínimo um dispositivo; e 2) cada dispositivo pode ser associado a apenas uma conta de usuário no Safe-Record. A identificação do dispositivo mob_i.id será um *hash* de seu UUID, o identificador unívoco

universal do aparelho, concatenado com uma sequência de *salting*. A ideia é permitir a identificação desse aparelho sem dar ao provedor o seu verdadeiro UUID, que poderia ser utilizado para rastrear o usuário em outras aplicações e, por fim, identificá-lo.

5.2.2 Autenticação

A autenticação é um elemento essencial de qualquer projeto de segurança. Além de ser o mecanismo pelo qual um usuário prova sua identidade ao sistema, também representa a primeira barreira entre a informação e um agente não autorizado. No Safe-Record um usuário deve, primeiramente, demonstrar conhecimento de um segredo pessoal - sua senha u_i.S. Para tanto, deverá executar o protocolo SRP. Em benefício da legibilidade, detalhes da comunicação trocada nesse processo foram omitidos neste capítulo, uma vez que foram expostos na sessão que trata do SRP no capítulo anterior.

Ao fim, se obtiver sucesso na execução do protocolo, o usuário será confrontado com outro desafio, um código de barras bidimensional no formato QR, que será usado para provar a posse de um dispositivo móvel mob_i, conhecido pelo sistema. O usuário usa o dispositivo para ler o código QR na tela. O dispositivo decodifica o *token*, produz uma assinatura digital e envia a resposta diretamente ao servidor. A aplicação na nuvem utiliza a chave RSA pública do dispositivo, mob_i.RSA.pk, para verificar a assinatura e, então, notifica o *browser* (a interface do aplicativo rodando na máquina do cliente) quanto ao sucesso da operação.

Existem métodos mais sofisticados de autenticação que introduzem um nível muito maior de segurança, como o uso de hardware criptográfico, ou ainda a técnica que entrega ao usuário uma lista de códigos ou um gerador de códigos aleatórios. Muitas soluções atuais de informática em saúde recorrem a *hardwares* criptográficos, como *smart cards* e *tokens*, para realizar o controle de acesso e até mesmo encriptar os registros clínicos. Entre elas, alguns projetos importantes, como o 'Cartão da Saúde' alemão e o TMT de Taiwan (Taiwan Electronic Medical Record Template) [77]. Mais uma vez, é preciso ponderar bem o uso de protocolos de autenticação complexos, evitando sempre que a perda de um hardware ou outro elemento físico chegue a impossibilitar o acesso a um prontuário médico.

Outra solução comumente adotada na literatura é estabelecer um serviço de identidade em um servidor confiável, preferencialmente fora da nuvem [81, 26]. No Safe-Record, o papel do 'servidor confiável' é desempenhado pelo dispositivo acreditado na conta do usuário.

Note que o Safe-Record requer um dispositivo comum, que o usuário já porta normalmente no seu cotidiano. Isso diminui a complexidade e o custo da solução. Também diminui o risco de que a operação normal do sistema seja afetada pela falta de um dispo-

sitivo. O risco de que esse dispositivo esteja ausente em um caso de emergência também é menor. De qualquer forma, como será explanado adiante, uma autorização de emergência pode ser criada por um gestor. Mesmo que perca o dispositivo e precise recorrer à autoridade credenciadora (o médico ou gestor que o credenciou no sistema) para habilitar um novo dispositivo, os dados permanecerão a salvo. Ademais, as chaves do usuário estão gravadas, em forma cifrada, na nuvem, o que reduz a zero a probabilidade de que a falta de um dispositivo específico venha a impedir, em definitivo, a leitura do registro.

Prosseguindo no processo de autenticação, ao fim dos dois desafios, o usuário receberá a chave RSA privada numa mensagem PEM cifrada no padrão PKCS#8, e um pacote encriptado com as chaves que dão acesso ao registro. A chave temporária pbekey é derivada a partir da senha e usada para decifrar o chave RSA privada do usuário. Essa chave privada é, então, usada para decifrar o pacote de chaves. Com todas as chaves em memória, o usuário pode realizar buscas, visualizar e alterar os registros.

Algoritmo 5.2. Autenticação

```
1: procedure Browser.Authenticate(u_i.id, u_i.S)
2:     u_i.R ← Cloud.RetrieveSalt(u_i.id)
3:     u_i.SRP.v ← SRP.Setup(u_i.R, u_i.S)
4:     if Cloud.SRP.Verify(u_i.id, u_i.SRP.) then
5:         token ← Cloud.GeneraeAuthChallenge()
6:         if Mobile.Authenticate(token) then
7:             pem ← Cloud.RetrivePKCS#8PEM(u_i.id)
8:             encapsuled ← Cloud.RetriveEncapsuledKeys(u_i.id)
9:             pbekey ← PBKDF2(u_i.S)
10:            u_i.RSA ← PKCS#8.Dec(pem, pbekey)
11:            u_i.keySet ← RSA.Dec(encapsulated, u_i.RSA.sk)
12:        end if
13:    else
14:        % Authentication fails
15:    end if
16: end procedure
17: procedure Mobile.Authenticate(token)
18:    response ← RSA.sign(token, mob_i.RSA.sk)
19:    if Cloud.VerifySignature(response) then
20:        % Authentication succeeds
21:    else
22:        % Authentication fails
23:    end if
24: end procedure
```

5.2.3 Autorização

O Safe-Record utiliza um sistema de autorização misto. Num primeiro estágio, o controle é criptográfico: somente os portadores da chave que cifra o registro têm acesso a ele. No segundo estágio, o controle é puramente lógico (embarcado na lógica da aplicação), e é baseado em papeis. Isto é, as operações que podem ser executadas sobre as informações serão limitadas de acordo com os papéis exercidos pelos usuários.

Contratos

Para que usuários diferentes (e.g. paciente e médicos) possam compartilhar a posse das chaves de um registro, foi instituído um protocolo de compartilhamento de chaves. Esse compartilhamento de chaves é referenciado como 'contrato' neste trabalho, pois representa a manifestação de vontade e também uma relação de confiança entre dois usuários. Por exemplo, ao visitar um clínico ou especialista, o paciente estabelece esta relação de confiança, concedendo acesso ao seu registro àquele profissional.

Em princípio, as chaves que cifram o registro do paciente são encapsuladas com sua própria chave RSA pública p_i.RSA.pk. Para ter acesso a elas, basta usar a chave privada p_i.RSA.sk para decifrar esse pacote de chaves. Esta chave privada, por sua vez, é encriptada usando sua senha do usuário. Por isso foi introduzida a exigência de que sempre exista pelo menos um contrato ativo para cada paciente. Pois, caso perca sua senha, o paciente necessita de uma cópia das chaves utilizadas em seu registro.

O contrato paciente-médico, não resulta, no entanto, no compartilhamento da chave RSA privada do paciente. Essa chave é utilizada para produzir as assinaturas digitais que provam a autoria do contrato e, portanto, deve permanecer sempre em estrita posse do paciente. O contrato consiste, então, no encapsulamento das chaves dos sistemas AES, AHE e OPE com a chave pública do médico em questão, acompanhado de uma assinatura digital sobre um certificado que atesta a identidade deste clínico. O Algoritmo 5.3 mostra os passos de criação de um contrato.

Algoritmo 5.3. Autorização

Require: usuário autenticado p_i

1: **procedure** BROWSER.CREATECONTRACT(d_j.id)
2: encapsuled ← RSA.Enc(p_i.keyset, d_j.RSA.pk)
3: cert ← X509v3.createCertficate(d_j.id)
4: signature ← RSA.Sign(cert, p_i.RSA.sk)
5: Cloud.StoreContract(p_i.id, d_j.id, encapsulated, signature)
6: **end procedure**

Todos esses dados são deletados quando o paciente revoga o contrato. O que pode ser feito por meio da funcionalidade de gestão de contratos disponível para todos os usuários. O paciente também será informado do fato de que o gestor daquele clínico tem acesso, ainda que indiretamente, a seu prontuário.

Autorizações de emergência

Para que um médico que nunca atendeu um paciente tenha acesso a seu prontuário em um caso de emergência, um gestor autorizado pode decifrar as chaves do paciente encriptadas com sua chave RSA pública e cifrá-las com a chave do médico que está prestando o atendimento emergencial. Esse contrato levará a assinatura digital do gestor no lugar da

assinatura do paciente e aparecerá com o devido destaque na lista de contratos daquele paciente. Como já apresentado anteriormente, uma vez recuperado, o paciente pode usar essa lista para revogar o contrato emergencial.

5.2.4 Análise

A análise da segurança do controle de acesso no Safe-Record é conduzida passo-a-passo, através dos procedimentos básicos explanados acima. A intenção é a demonstrar como as primitivas criptográficas empregadas garantem as propriedades de segurança exigidas em cada ponto de interação.

Credenciamento:

Quando um usuário é credenciado, o servidor na nuvem recebe apenas a sequência aleatória u_i.R, o verificador a ser usado no protocolo SRP u_i.SRP.v, a chave RSA pública u_i.RSA.pk, uma chave encriptada no formato PEM (PKCS#8 v2.0) e um bloco encriptado sob o sistema RSA-OAEP, que contém as chaves que cifram os registros privados daquele usuário.

O provedor curioso não tem acesso a qualquer bit significativo do conteúdo do pacote cifrado e, a menos que seja capaz de quebrar o sistema RSA-OAEP, não terá acesso ao conteúdo desse bloco. A segurança desse criptossistema já foi suficientemente analisada na literatura, e as implementações em bibliotecas padrão já sofreram o necessário escrutínio para aplicações reais [12, 45, 23]. Portanto, nessa análise, essa cifra é considerada segura, por definição, nos quesitos IND-CPA (indistinguível sob um ataque de texto escolhido) e IND-CCA (indistinguível sob um ataque de texto cifrado escolhido) e IND-CCA2 (seguro sob ataque de texto-cifrado escolhido adaptativo). Isso quer dizer que, mesmo conhecendo a estrutura interna do bloco encriptado, mesmo tendo algum conhecimento da operação do sistema, e mesmo podendo usá-lo para cifrar um número arbitrário de amostras, o atacante não obterá qualquer vantagem na criptoanálise dos dados cifrados por outros usuários.

O atacante também não terá acesso à chave RSA encapsulada no formato PEM, a menos que consiga quebrar a cifra AES-256-CBC usada no algoritmo PBES2 (uma subrotina do encapsulamento PKCS#8). O sistema AES, assim como o RSA, é um sistema maduro, as implementações de referência também sofreram o devido crivo, tanto na academia quanto na indústria, e, portanto, será simplesmente aceito como seguro nessa análise [88].

A dificuldade de recuperar a senha do usuário a partir do verificador u_i.SRP.v persistido na nuvem é semelhante àquela do problema decisional de Diffie-Hellman, pois

consiste na computação do logaritmo discreto no campo finito $\mathbb{Z}_P$, onde P é um primo grande (1024 bits) [121]. Portanto, o atacante também não poderá fazer uso dos elementos de credenciamento para recuperar a senha ou roubar a identidade do usuário. Mesmo que fosse capaz de aprender a senha por outros meios, não teria acesso aos dados sem apresentar um dispositivo acreditado pelo usuário.

Autenticação:

A prova de identidade derivada durante a execução do protocolo SRP (uma chave compartilhada com entropia forte) não pode ser produzida sem o conhecimento da senha u_i.S, e a computação de u_i.SRP.v se dá no lado do cliente, durante o registro do usuário. Dessa forma, a senha do usuário nunca é exposta, seja em trânsito ou em armazenamento na nuvem.

A execução do protocolo SRP nas interações de autenticação, permite que a identidade seja verificada sem que o provedor jamais receba informações suficientes para recuperar a senha. A sequência aleatória u_i.R utilizada nesse processo é pública (i.e. está gravada na nuvem). No entanto, as propriedades do protocolo SRP garantem que o conhecimento desse dado não altera as chances do provedor de recuperar a senha.

A autenticação é exigida tanto no *front-end*, rodando no cliente, quanto no *back-end* rodando na nuvem, de forma que nenhuma informação será exposta, nem mesmo na forma encriptada, sem que o usuário tenha primeiramente vencido os desafios de autenticação. Outro mecanismo de proteção no *back-end* é o controle de tentativas de autenticação, que recusa quaisquer requisições de um dispositivo após um dado número de falhas na autenticação.

Após a autenticação, a senha do usuário será utilizada para derivar a chave temporária pbekey, que decifra a chave RSA privada u_i.RSA.sk. Essa chave RSA, por sua vez, é usada para decifrar o pacote com as demais chaves. Nesse momento, como mostra a Figura 5.2, todas as chaves estão na memória do dispositivo do usuário e um ataque ativo do provedor envolvendo a injeção de código no *browser* poderia expor todas elas.

Isso não ocorre no Safe-Record, pois o sistema não disponibiliza qualquer funcionalidade, nem mesmo a autenticação, se o usuário não tiver a extensão BrowserTrust instalada no navegador. Essa extensão realiza a verificação do código entregue pelo provedor, bem como a execução de todas as funções criptográficas. O Safe-Record também utiliza a funcionalidade de CSP (Content Security Policy) do *browser* para impedir que qualquer código *in-line*, ou seja, embarcado no HTML, seja executado [118]. Também usa a extensão SRI (Sub-Resource Integrity) como uma primeira linha de defesa contra alterações no código entregue pelo provedor [1].

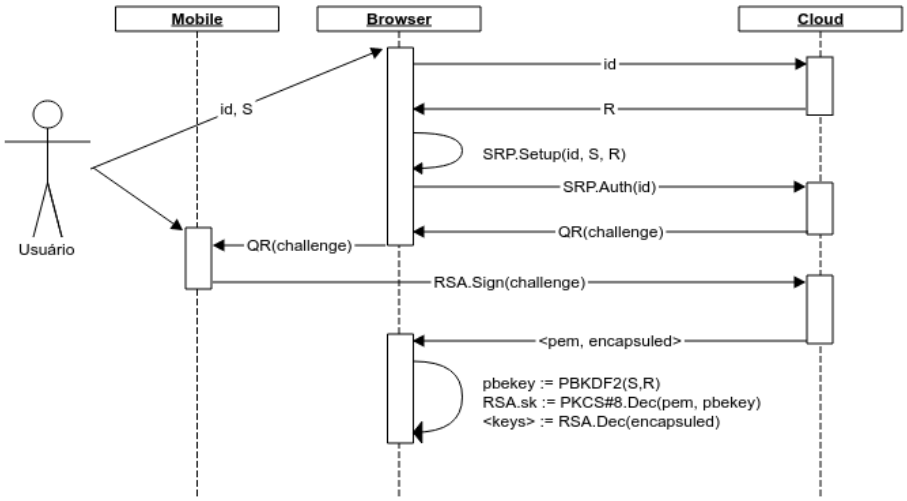

Figura 5.2: Autenticação por dois fatores

Quando o navegador inicia uma requisição para uma URL que identifica uma das instâncias do CloudEHR rodando na nuvem, a extensão dispara um processo de verificação e observa os recursos externos carregados (*scripts*, folhas de estilo e imagens), calcula o *hash* SHA384 desses recursos (de acordo com o padrão SRI [1]) e confronta com uma lista embarcada numa área de memória interna da extensão. Caso algum arquivo apresente conteúdo diverso do esperado, o usuário é alertado com uma mensagem exibida na tela e, a partir daí, o BrowserTrust não aceitará mais nenhuma requisição daquela fonte.

A extensão responde a muitas requisições do *front-end* porque também absorveu todas as funções criptográficas. Cifragem, decifragem, derivação de chaves, *hash* e até mesmo a execução do protocolo SRP se dão na extensão. Dessa forma, todas as chaves criptográficas são mantidas na área de memória da extensão, que roda como um processo separado do navegador. No entanto, o *sand-boxing*, ou isolamento, desse processo é controlado pelo próprio *browser* e pode ser comprometido se o usuário estiver rodando uma versão 'não-padrão' do navegador, ou se tiver habilitado extensões com código malicioso. Então, a extensão BrowserTrust estará segura desde que o usuário tenha uma instalação padrão, limpa, do *browser*. Se a extensão rodar como esperado, então o *front-end* e as chaves criptográficas estarão seguros.

Se o *front-end* estiver seguro, então o controle de autenticação não será quebrado, a sessão do usuário não poderá ser roubada – pois a chave da sessão, derivada no protocolo SRP, estará em uma área protegida de memória. Essa chave é usada para autenticar os *tokens* usados na mitigação de ataques CSRF (desafios renovados a cada requisição ao servidor).

Autorização:

A única maneira de acessar um registro é ter um contrato que o autorize. O contrato não pode ser forjado, a menos que o atacante possa forjar uma assinatura RSA-SSP. Assim como se deu com os sistemas RSA-OAEP e AES-256-CBC, essa análise assume que o sistema RSA-SSP é provadamente seguro [13].

Mesmo que pudesse forjar a assinatura, o atacante teria, no máximo, acesso a um pacote encriptado com uma chave RSA que ele não possui. Para vencer mais essa barreira, o atacante necessita quebrar a cifra AES-256-CBC (usada no encapsulamento PBES2, do padrão PKCS#8) e, assim ganhar acesso à chave RSA privada necessária decifrar as chaves de um registro. A Figura 5.3 mostra a criação de um contrato de acesso por parte de um paciente, bem como sua utilização por parte do médico autorizado.

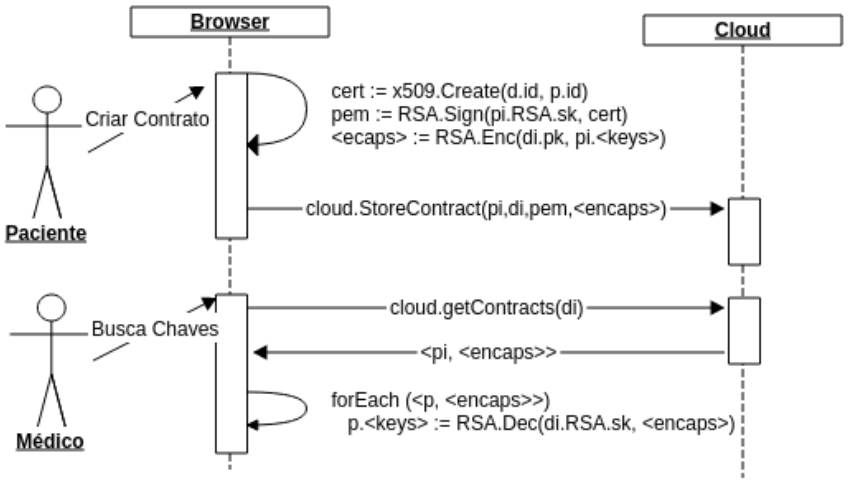

Figura 5.3: Contratos de autorização de acesso

Como demonstrado acima, uma vez obedecidos os requisitos de segurança da extensão rodando no navegador, um atacante também não será capaz de acessar indevidamente os dados decifrados no cliente. Dessa forma, é possível afirmar que, observados os requisitos mínimos de funcionamento, quebrar o mecanismo de autorização do *Safe-Record* é tão difícil quanto quebrar o sistema RSA-SSP e o sistema AES-256-CBC juntos.

Uma preocupação válida que pode surgir neste ponto é o uso do mesmo par de chaves RSA tanto para a cifra do pacote de chaves quanto para a produção de assinaturas RSA-PSS. A segurança dessa combinação pode ser demostrada por meio da estratégia definida no trabalho de Haber e Pinkas [54]: demonstrar, primeiramente, que a presença de assinaturas digitais geradas com uma chave privada não afeta a segurança do sistema criptográfico que usa o par correspondente; e, então, demonstrar que a observação de

crifras geradas com uma chave pública não gera nenhuma vantagem para um eventual atacante contra o sistema de assinatura.

Sejam:

$\mathcal{A}$ Um atacante, definido como um algoritmo PPT (probabilistico em tempo polinomial);

ES Um esquema criptográfico assimétrico, composto pelos algoritmos PPT KeyGen, Enc e Dec;

SS Um esquema de assinaturas assimétrico, composto pelos algoritmos PPT KeyGen, Sign e Verify.

KeyGen Algoritmo gerador do par de chaves $\langle \mathrm{pk}, \mathrm{sk} \rangle$, de cifragem/decifragem e assinatura/verificação;

Enc Algoritmo de cifragem por chave pública, que, dadas a chave pública pk e uma mensagem $\mathrm{m} \in \{0,1\}^*$, computa a cifra $\mathrm{c} \in \{0,1\}*$;

Dec Algoritmo de decifragem por chave privada, que, dadas a chave privado sk e uma cifra $\mathrm{c} \in \{0,1\}^*$, apresenta a mensagem correspondente $\mathrm{m} \in \{0,1\}^*$ ou uma mensagem de erro;

Sign Algoritmo de assinatura digital por chave privada que, ao receber a chave privada sk e uma mensagem $\mathrm{m} \in \{0,1\}^*$, produz a assinatura $\sigma \in \{0,1\}^*$;

Verify Algoritmo de verificação da assinatura, com uso de chave pública que, ao receber a chave pública pk e o par mensagem/assinatura (m, σ), responde 1, em caso de aceitação, ou 0, caso contrário.

Dados ES = KeyGen, Enc, Dec) e SS = (KeyGen, Sign, Verify), podemos afirmar que o uso combinado de ES|SS não afeta a segurança de ES se, e somente se, para qualquer adversário $\mathcal{A}$ existir um adversário $\mathcal{A}'$, contra o esquema ES isoladamente, com probabilidade de sucesso igual ou com diferença ε desprezível em relação à probabilidade de sucesso de $\mathcal{A}$. Chamamos o desafio de $\mathcal{A}$ contra o sistema combinado ES|SS $\mathrm{PubK}^{\mathrm{CPA}}_{\mathcal{A},\mathrm{ES|SS}}$, e o de $\mathcal{A}'$ contra o sistema ES de $\mathrm{PubK}^{\mathrm{CPA}}_{\mathcal{A}',\mathrm{ES}}$. O sistema é considerado seguro se a seguinte equação puder ser provada.

$$\Pr[\mathrm{PubK}^{\mathrm{CPA}}_{\mathcal{A},\mathrm{ES|SS}} = 1] = \Pr[\mathrm{PubK}^{\mathrm{CPA}}_{\mathcal{A}',\mathrm{ES}}] + \varepsilon \tag{5.1}$$

Para provar a segurança do sistema combinado ES|SS, podemos imaginar um simulador nos seguintes moldes:

Setup o simulador gera o par de chaves $\langle \mathrm{pke}, \mathrm{sk} \rangle = \mathrm{KeyGen}(1^{\mathrm{k}})$ e entrega a chave pk ao adversário $\mathcal{A}$;

Fase 1 o adversário gera um número limitado de decifragens (o simulador age como oráculo de deciframente) e um número limitado de assinaturas (oráculo de assinatura);

Desafio O adversário $\mathcal{A}$ produz duas mensagens $\mathrm{m}_0, \mathrm{m}_1$, para a qual o simulador responde com $\mathrm{c} = \mathrm{Enc}(\mathrm{pk}, \mathrm{m}_{\mathrm{b}})$, onde b é tomando uniformemente em $\{0, 1\}$;

Fase 2 o adversário realiza uma nova rodada de consultas ao oráculo de deciframento;

Teste o adversário apresenta sua decisão acerca de b.

O trabalho de Shoup [101] traz uma extensa prova de que, em decorrência das propriedades algébricas da construção RSA-OAEP, a probabilidade de sucesso do adversário nesse experimento com a fase um é exatamente igual à sua probabilidade de sucesso sem esta fase. Logo, a probabilidade de sucesso de $\mathcal{A}$ será sempre igual à vantagem de $\mathcal{A}'$. Portanto, a presença de um oráculo de assinatura RSA-PSS não interfere na segurança da cifra RSA-OAEP.

5.3 Cifra e manipulação dos registros clínicos

O mecanismo de controle de acesso descrito na seção anterior é um elemento crucial para a segurança dos registros clínicos armazenados na nuvem. Uma vez que a política de controle de acesso é imposta, resta analisar os demais requisitos da aplicação.

Nesta sessão, o projeto do das funcionalidades de gestão dos registros clínicos mostra como os criptossistemas apresentados no capítulo anterior são usados para permitir a realização de buscas sobre os registros na nuvem.

5.3.1 Encriptação

Os objetos que representam instâncias dos modelos de dados no cliente tem atributos que indicam o tipo correspondente no pacote de Data Types, do OpenEHR – detalhado no Capítulo 3, deste trabalho. Quando identifica uma entrada de um dos tipos litados a seguir, a camada de modelo do *front-end* requisita ao serviço de criptografia (na extensão do navegador) que aquela entrada seja cifrada com os sistema AHE de Paillier e com o sistema OPE de Boldyreva:

- DV_STATE e DV_IDENTIFIER, do pacote Basic
- DV_QUANTITY e DV_COUNT do pacote Quantity;

- DV_DATE, DV_TIME, DV_DATE_TIME, do pacote Date Time;
- DV_CODED_TEXT e CODE_PHRASE, do pacote Text;

Entradas do tipo CODE_PHRASE, na realidade, são substituídas por um índice de uma lista de domínios, uma tabela com a lista de identificadores internos nas ontologias embarcadas nos arquétipos implementados. O tipo CODE_PHRASE compõe a classe DV_CODED_TEXT, que, por sua vez, é um atributo comum a quase todas as demais classes de entradas de dados num arquétipo, pois é utilizado para identificá-las. O tipo CODE_PHRASE também é utilizado como base para os tipos DV_STATUS e DV_IDENTIFIER, do pacote Basic,

As demais entradas no arquétipo são cifradas apenas com o sistema AES-256-GCM. Na verdade, o registro como um todo, no formato XML, é usado como entrada para a cifra AES. O resultado desse ciframento é considerado o formato original do arquétipo no prontuário do paciente. As demais cifras têm papel secundário: são índices para a identificação e seleção de registros, ou são fonte de informação anônima (médias, modas, etc) sobre todas as coleções de registros, nos diversos prontuários. A Figura 5.4 apresenta essa estratégia de leitura/alteração dos registros.

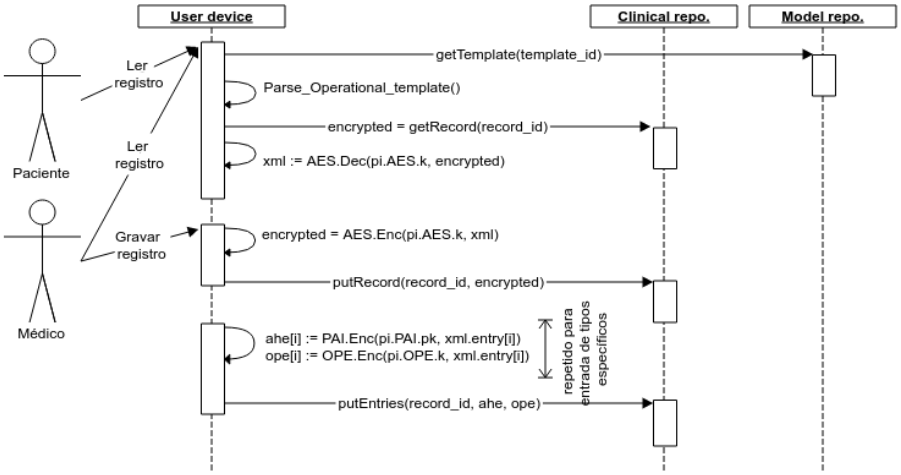

Figura 5.4: Gestão de registros

5.3.2 Buscas

Como discutido anteriormente, apenas um conjunto mínimo de arquétipos foi implementado no protótipo construído neste trabalho. Os arquétipos selecionados são aqueles referenciados pelo *template* operacional **RAS-AB** (Resumo de Atendimento em Saúde - Atenção Básica), usado como padrão no âmbito da rede de atenção básica do Sistema Único de Saúde. Essa rede conta atualmente com dois sistemas principais: o e-SUS-AB, uma coleção de aplicativos usados nas unidades de atendimento para registrar diversos tipos de atendimento à população (clínica, dentária, campanhas de vacinação etc.); e o SISAB (Sistema de Informação em Saúde para a Atenção Básica), o sistema de informação nacional, que recebe registros das aplicações e-SUS-AB em um formato de exportação próprio.

Esses sistemas estão passando por uma larga reformulação, que visa melhorar a qualidade da informação que dá suporte à elaboração de políticas públicas nacionais em saúde. Entre os objetivos, estão eliminar duplicidades, construir um histórico clínico consistente de cada paciente, construir uma base de dados que permita a execução de medidas estatísticas mais realistas e precisas. Um dos elementos dessa estratégia é a adoção do OpenEHR como padrão de modelagem de dados. O *template* utilizado nesta pesquisa resulta justamente de um trabalho de cooperação técnica entre pesquisadores deste Departamento com os administradores do SISAB no Ministério da Saúde.

O arquivo de definição do RAS-AB (Apêndice A - RAS-AB.oet) contém entradas definidas em 46 arquétipos. Alguns oriundos do repositório público do OpenEHR e outros criados pelos especialistas do Ministério. Entre eles estão os arquétipos openEHR-EHR-OBSERVATION.height.v1 e openEHR-EHR-OBSERVATION.body_weight.v1. Ambos são compostos por uma entrada da classe DV_CODED_TEXT, que representa o tipo da medida (altura ou peso corporal), e uma entrada da classe DV_QUANTITY, que representa a medida em si.

O tipo DV_CODED_TEXT tem um atributo CODE_PHRASE que representa o código de um termo dentro de uma ontologia. Na tela, esse tipo é exibido como uma etiqueta, informando ao usuário o significado da próxima entrada no arquétipo. Internamente, no entanto, o atributo CODE_PHRASE é um código numérico, um índice tomado de uma tabela que lista os identificadores internos nas ontologias embarcadas nos arquétipos utilizados neste protótipo. Ou seja, os 46 arquétipos referenciados no *template* operacional implementado.

A classe CODE_PHRASE também é utilizada como atributo nas classes DV_STATUS e DV_IDENTIFIER, do pacote Basic. Dessa forma, para executar uma busca sobre os registros e selecionar aqueles que apresentam uma determinada entrada, basta cifrar o

CODE_PRASE, o código identificador dessa entrada, com o sistema com preservação de ordem (OPE).

Após identificar os registros que possuam, por exemplo, a observação de pressão sanguínea sistólica (definida no arquétipo openEHR-EHR-OBSERVATION.blood_pressure.v1), é possível refinar a busca, e apresentar apenas aquelas observações superiores a 100 mmHg. Para isso, é preciso cifrar o número 100 com o sistema OPE e comparar as magnitudes. Essa busca deve ser preparada no cliente, isto é, as operações de cifra acontecem na máquina do usuário, onde as chaves criptográficas estão. As comparações são feitas na nuvem e as cópias cifradas com o esquema AES-256-GCM dos registros selecionados são enviadas para a máquina do cliente, onde são decifradas e exibidas. Esse sistema é extremamente eficiente. Dessa forma, esse último passo da busca tem impacto desprezível na velocidade final da busca.

Em todos os arquétipos apresentados como exemplo acima, a magnitude das entradas de classe DV_QUANTITY é cifrada também com o sistema aditivamente homomórfico. Pelas propriedades do sistema AHE utilizado, o provedor da nuvem utiliza a chave pública para realizar a soma dos valores cifrados, e produz um resultado cifrado, por meio do qual não poderá conhecer o valor da soma. O resultado deve ser decifrado na máquina do usuário, onde estará a chave privada. O provedor da nuvem deve enviar a soma cifrada e o número de observações. O *front-end*, então, decifra a soma e divide pelo número de observações, obtendo assim a média.

Essas duas estratégias de consulta pode ser vistas na Figura 5.5.

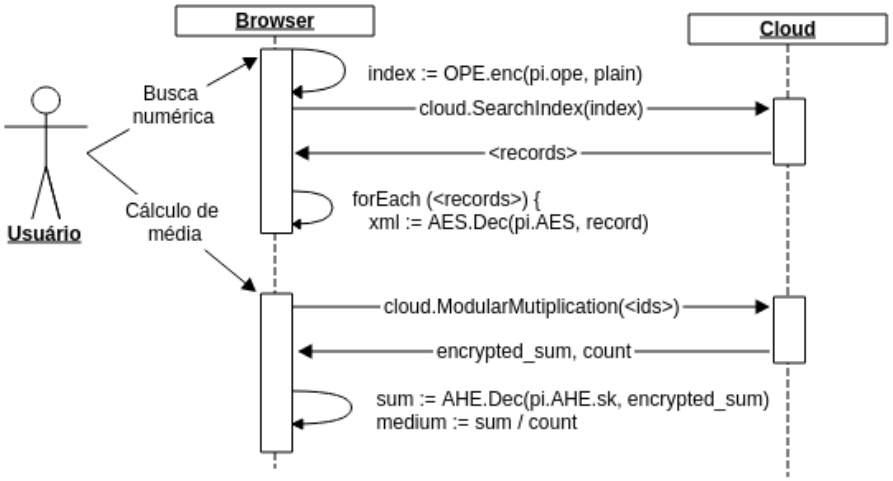

Figura 5.5: Busca numérica e cálculo de média

Os tipos de consultas e as estratégias usadas para realizá-las dependem do arquétipo ou *template* implementado. Essas estratégias manualmente construídas para cada arqué-

tipo resultam em uma busca mais eficiente. É claro que esta arquitetura implica uma participação muito maior do cliente na preparação e computação das buscas do que em um sistemas que envie os registros em claro para nuvem. Além disso, nenhum requisito especial é imposto sobre o hardware do cliente. Portanto, a solução terá, por projeto, um desempenho mais pobre do que o de sistemas comuns. Os ganhos em segurança e privacidade, no entanto, mais que compensam a perda de desempenho.

5.3.3 Análise

Para analisar a segurança do sistema resultante, primeiramente, precisamos considerar a segurança da combinação do sistma RSA-OAPE, utilizado no primeiro passo do acesso ao dado (acesso à chave) e o sistema AES-256-GCM, correspondente ao segundo passo (acesso ao registro). Como discutido no Capítulo 4, a única exigência para que o sistema AES-256-GCM seja CPA-seguro é que os *nonces* introduzidos na cifra de cada registro não se repitam. Esse requisito é satisfeito pelo simples uso do identificador unívoco do registro na tabela relacional como entrada para a função geradora de *nonces* [34].

Sejam Π = RSA-OAEP e Π' = AES-GCM-256, para provar a segurança do sistema híbrido $\Pi^{\text{hy}} = (\Pi, \Pi')$ é CCA-seguro, se ambos os esquemas que o compoem forem CCA-seguros.

A prova consiste em 4 passos, seguindo o seguinte esquema, onde as setas indicam a indistinguibilidade ou segurança contra ataques de texto cifrado escolhido.

$$
\begin{array}{ccc}
 & \text{(por transitividade)} & \\
(\text{pk}, \text{Enc}_{\text{pk}}(\text{k}), \text{Enc}'_{\text{k}}(\text{m}_0)) & \dashleftarrow\!\dashrightarrow & (\text{pk}, \text{Enc}_{\text{pk}}(\text{k}), \text{Enc}'_{\text{k}}(\text{m}_1)) \\
 & & \\
(\text{por segurança de } \Pi) \updownarrow & & \updownarrow \text{ (por segurança de } \Pi) \\
 & & \\
(\text{pk}, \text{Enc}_{\text{pk}}(0^{\text{n}}), \text{Enc}'_{\text{k}}(\text{m}_0)) & \longleftrightarrow & (\text{pk}, \text{Enc}_{\text{pk}}(0^{\text{n}}), \text{Enc}'_{\text{k}}(\text{m}_1)) \\
 & \text{(por segurança de } \Pi' \text{)} &
\end{array}
$$

Desejamos mostrar que no experimento $\text{PubK}^{\text{cca}}_{\mathcal{A}^{\text{hy}},\Pi^{\text{hy}}(1^{\text{n}})}$, que testa a indistinguibilidade do esquema Π^{hy} diante de um ataque de texto cifrado escolhido, qualquer adversário $\mathcal{A}^{\text{hy}}$ probabilistico e de tempo polinomial tem vantagem desprezível na tentativa de discernir entre uma mensagem cifrada pelo esquema Π^{hy} e um elemento qualquer do espaço de textos cifrados. Ou seja, existe uma função desprezível negl tal que:

$$\Pr[\text{PubK}^{\text{cca}}_{\mathcal{A}^{\text{hy}},\Pi^{\text{hy}}(1^{\text{n}})} = 1] \leq \frac{1}{2} + \text{negl}(\text{n}) \tag{5.2}$$

Considerando a distribuição de probabilidade sobre a escolha de b, pela definição do experimento, podemos dizer que:

$$\begin{aligned}\Pr[\mathrm{PubK}^{\mathrm{cca}}_{\mathcal{A}^{\mathrm{hy}},\Pi^{\mathrm{hy}}}(1^{\mathrm{n}}) = 1] = \frac{1}{2}\Pr[\mathcal{A}^{\mathrm{hy}}\langle \mathrm{Enc}_{\mathrm{pk}}(\mathrm{k}), \mathrm{Enc}'_{\mathrm{k}}(\mathrm{m}_0)\rangle = 0] \\ + \frac{1}{2}\Pr[\mathcal{A}^{\mathrm{hy}}\langle \mathrm{Enc}_{\mathrm{pk}}(\mathrm{k}), \mathrm{Enc}'_{\mathrm{k}}(\mathrm{m}_1)\rangle = 1]\end{aligned} \tag{5.3}$$

Passo I. Experimento $\mathrm{PubK}^{\mathrm{cca}}_{\mathcal{A}_1,\Pi(1^{\mathrm{n}})}$:

1.1. $\mathcal{A}_1$ recebe pk, escolhe $\mathrm{k} \leftarrow \{0,1\}^{\mathrm{n}}$ aleatoriamente (n pode ser obtido de pk) e apresenta como saída o par de mensagens $(\mathrm{k}, 0^{\mathrm{n}})$;

1.2. $\mathcal{A}_1$ recebe o texto cifrado c_1 e envia pk para $\mathcal{A}^{\mathrm{hy}}$, que responde com duas mensagens de igual tamanho $\mathrm{m}_0, \mathrm{m}_1$;

1.3. $\mathcal{A}_1$ então calcula $\mathrm{c}_2 := \mathrm{Enc}'_{\mathrm{k}}(\mathrm{m}_0)$ e envia $(\mathrm{c}_1, \mathrm{c}_2)$ para $\mathcal{A}^{\mathrm{hy}}$ que, por sua vez, apresenta como saida o bit b';

1.4. $\mathcal{A}_1$ apresenta como saida o bit b' (que foi gerado por $\mathcal{A}^{\mathrm{hy}}$).

Quando $\mathrm{b} = 0$ no experimento $\mathrm{PubK}^{\mathrm{cca}}_{\mathcal{A}_1,\Pi(1^{\mathrm{n}})}$, o adversario $\mathcal{A}_1$ recebe um texto cifrado c_1 da forma $\mathrm{Enc}_{\mathrm{pk}}(\mathrm{k})$, onde k foi escolhido aleatoriamente por $\mathcal{A}_1$ no passo 1.1. Isso significa que $\mathcal{A}^{\mathrm{hy}}$ recebe um texto cifrado $(\mathrm{c}_1, \mathrm{c}_2) = (\mathrm{Enc}_{\mathrm{pk}}(\mathrm{k}), \mathrm{Enc}'_{\mathrm{k}}(\mathrm{m}_0))$. Assim:

$$\Pr[\mathrm{Output}(\mathcal{A}_1) = 0 | \mathrm{b} = 0] = \Pr[\mathcal{A}^{\mathrm{hy}}\langle \mathrm{Enc}_{\mathrm{pk}}(\mathrm{k}), \mathrm{Enc}'_{\mathrm{k}}(\mathrm{m}_0)\rangle = 0] \tag{5.4}$$

Por outro lado, quando $\mathrm{b} = 1$ no experimento $\mathrm{PubK}^{\mathrm{cca}}_{\mathcal{A}_1\Pi(1^{\mathrm{n}})}$, o adversario $\mathcal{A}_1$ recebe um texto cifrado c_1 da forma $\mathrm{Enc}_{\mathrm{pk}}(0^{\mathrm{n}})$. Nesse caso, $\mathcal{A}^{\mathrm{hy}}$ recebe um texto cifrado da forma $(\mathrm{c}_1, \mathrm{c}_2) = (\mathrm{Enc}_{\mathrm{pk}}(0^{\mathrm{n}}), \mathrm{Enc}'(\mathrm{m}_0))$. Assim:

$$\Pr[\mathrm{Output}(\mathcal{A}_1) = 1 | \mathrm{b} = 1] = \Pr[\mathcal{A}^{\mathrm{hy}}(\mathrm{Enc}_{\mathrm{pk}}(0^{\mathrm{n}}), \mathrm{Enc}'_{\mathrm{k}}(\mathrm{m}_0)) = 1] \tag{5.5}$$

Como demonstrado na seção anterior (e em [12, 45, 23, 101], o sistema RSA-OAEP é é CCA-seguro, logo, existe uma função desprezível negl_1 tal que:

$$\begin{aligned}\frac{1}{2} + \mathrm{negl}_1(\mathrm{n}) &\geq \Pr[\mathrm{PubK}^{\mathrm{cca}}_{\mathcal{A}_1,\Pi(1^{\mathrm{n}})} = 1] \\ &= \frac{1}{2}(\Pr[\mathrm{Output}(\mathcal{A}_1) = 0 | \mathrm{b} = 0] + \Pr[\mathrm{Output}(\mathcal{A}_1) = 1 | \mathrm{b} = 1]) \\ &= \frac{1}{2}\Pr[\mathcal{A}^{\mathrm{hy}}\langle \mathrm{Enc}_{\mathrm{pk}}(\mathrm{k}), \mathrm{Enc}'_{\mathrm{k}}(\mathrm{m}_0)\rangle = 0] \\ &+ \frac{1}{2}\Pr[\mathcal{A}^{\mathrm{hy}}\langle \mathrm{Enc}_{\mathrm{pk}}(0^{\mathrm{n}}), \mathrm{Enc}'_{\mathrm{k}}(\mathrm{m}_0)\rangle = 1]\end{aligned} \tag{5.6}$$

Passo II. Experimento $\mathrm{PrivK}^{\mathrm{cca}}_{\mathcal{A}_2,\Pi'(1^n)}$:

2.1. $\mathcal{A}_2$ roda $\mathrm{Gen}(1^n)$ com o fim de gerar as chaves $(\mathrm{pk}, \mathrm{sk})$;

2.2. $\mathcal{A}_2$ informa (pk) a $\mathcal{A}^{\mathrm{hy}}$, que responde com duas mensagens $\mathrm{m}_0, \mathrm{m}_1$;

2.3. $\mathcal{A}_2$, recebe um texto cifrado c_2, calcula $\mathrm{c}_1 := \mathrm{Enc}_{\mathrm{pk}}(0^n)$ e então envia $(\mathrm{c}_1, \mathrm{c}_2)$ para $\mathcal{A}^{\mathrm{hy}}$, que, por sua vez, apresenta como saida o bit b'

2.4. $\mathcal{A}_2$ apresenta como resposta o bit b' que foi gerado por $\mathcal{A}^{\mathrm{hy}}$

Quando $\mathrm{b} = 0$ no experimento $\mathrm{PrivK}^{\mathrm{cca}}_{\mathcal{A}_2,\Pi'(1^n)}$, o adversario $\mathcal{A}_2$ recebe um texto cifrado da forma $\mathrm{Enc}'_{\mathrm{k}}(\mathrm{m}_0)$, onde k foi escolhido aleatoriamente (e é desconhecido de por $\mathcal{A}_2$). Isso significa que $\mathcal{A}^{\mathrm{hy}}$ recebe um texto cifrado $(\mathrm{c}_1, \mathrm{c}_2) = (\mathrm{Enc}_{\mathrm{pk}}(0^n), \mathrm{Enc}'_{\mathrm{k}}(\mathrm{m}_0))$. Assim:

$$\Pr[\mathrm{Output}(\mathcal{A}_2) = 0 | \mathrm{b} = 0] = \Pr[\mathcal{A}^{\mathrm{hy}}\langle \mathrm{Enc}_{\mathrm{pk}}(0^n), \mathrm{Enc}'_{\mathrm{k}}(\mathrm{m}_0)\rangle = 0] \tag{5.7}$$

Por outro lado, quando $\mathrm{b} = 1$ no experimento $\mathrm{PrivK}^{\mathrm{cca}}_{\mathcal{A}_2,\Pi'(1^n)}$, o adversario $\mathcal{A}_2$ recebe um texto cifrado da forma $\mathrm{Enc}'(\mathrm{m}_1)$. Isso significa que $\mathcal{A}^{\mathrm{hy}}$ recebe um texto cifrado da forma $(\mathrm{c}_1, \mathrm{c}_2) = (\mathrm{Enc}_{\mathrm{pk}}(0^n), \mathrm{Enc}'(\mathrm{m}_1))$. Assim:

$$\Pr[\mathrm{Output}(\mathcal{A}_2) = 1 | \mathrm{b} = 1] = \Pr[\mathcal{A}^{\mathrm{hy}}\langle \mathrm{Enc}_{\mathrm{pk}}(0^n), \mathrm{Enc}'_{\mathrm{k}}(\mathrm{m}_1)\rangle = 1] \tag{5.8}$$

Uma vez que, por hipótese, o sistema de encriptação de chave privada Π' é CCA-seguro, existe uma função desprezível negl_2 tal que:

$$\begin{aligned}
\frac{1}{2} + \mathrm{negl}_2(\mathrm{n}) &\geq \Pr[\mathrm{PrivK}^{\mathrm{cca}}_{\mathcal{A}_2,\Pi'(1^n)} = 1] \\
&= \frac{1}{2}(\Pr[\mathrm{Output}(\mathcal{A}_2) = 0 | \mathrm{b} = 0] + \Pr[\mathrm{Output}(\mathcal{A}_2) = 1 | \mathrm{b} = 1]) \\
&= \frac{1}{2}\Pr[\mathcal{A}^{\mathrm{hy}}\langle \mathrm{Enc}_{\mathrm{pk}}(0^n), \mathrm{Enc}'_{\mathrm{k}}(\mathrm{m}_0)\rangle = 0] \\
&+ \frac{1}{2}\Pr[\mathcal{A}^{\mathrm{hy}}\langle \mathrm{Enc}_{\mathrm{pk}}(0^n), \mathrm{Enc}'_{\mathrm{k}}(\mathrm{m}_1)\rangle = 1]
\end{aligned} \tag{5.9}$$

Passo III. Experimento $\mathrm{PubK}^{\mathrm{cca}}_{\mathcal{A}_3,\Pi(1^n)}$:

3.1. $\mathcal{A}_3$ recebe pk, escolhe $\mathrm{k} \leftarrow \{0,1\}^n$ aleatoriamente e apresenta como saída o par de mensagens $(0^n, \mathrm{k})$ (note a inversão da ordem em relação ao Passo I);

3.2. $\mathcal{A}_3$ recebe o texto cifrado c_1 e envia pk para $\mathcal{A}^{\mathrm{hy}}(\mathrm{pk})$, que responde com duas mensagens $(\mathrm{m}_0, \mathrm{m}_1)$;

3.3. $\mathcal{A}_3$ então calcula $\mathrm{c}_2 := \mathrm{Enc}'_{\mathrm{k}}(\mathrm{m}_1)$ e envia $(\mathrm{c}_1, \mathrm{c}_2)$ para $\mathcal{A}^{\mathrm{hy}}$;

3.4. $\mathcal{A}_3$ apresenta como saida o bit b' que foi gerado por $\mathcal{A}^{hy}$.

No experimento $\text{PubK}^{cca}_{\mathcal{A}_3,\Pi(1^n)}$, quando $b = 0$, o adversario $\mathcal{A}^{hy}$ recebe um texto cifrado na forma $(c_1, c_2) = (\text{Enc}_{pk}(0^n), \text{Enc}'_k(m_1))$, gerados a partir das chaves pk e k aleatórias. Assim:

$$\Pr[\text{Output}(\mathcal{A}_3) = 0|b = 0] = \Pr[\mathcal{A}^{hy}\langle \text{Enc}_{pk}(0^n), \text{Enc}'_k(m_1)\rangle = 0] \tag{5.10}$$

Por outro lado, quando $b = 1$ no experimento $\text{PubK}^{cca}_{\mathcal{A}_3,\Pi(1^n)}$, o adversario $\mathcal{A}^{hy}$ recebe um texto cifrado da forma $(c_1, c_2) = (\text{Enc}_{pk}(k), \text{Enc}'(m_1))$. Assim:

$$\Pr[\text{Output}(\mathcal{A}_3) = 1|b = 1] = \Pr[\mathcal{A}^{hy}\langle \text{Enc}_{pk}(k), \text{Enc}'_k(m_1)\rangle = 1] \tag{5.11}$$

Tendo em vista que, por hipótese, o sistema de encriptação de chave pública Π é CCA-seguro, existe uma função desprezível negl_3 tal que:

$$\begin{aligned}
\frac{1}{2} + \text{negl}_3(n) &\geq \Pr[\text{PubK}^{cca}_{\mathcal{A}_3,\Pi(1^n)} = 1] \\
&= \frac{1}{2}(\Pr[\text{Output}(\mathcal{A}_3) = 0|b = 0] + \Pr[\text{Output}(\mathcal{A}_3) = 1|b = 1]) \\
&= \frac{1}{2}\Pr[\mathcal{A}^{hy}\langle \text{Enc}_{pk}(0^n), \text{Enc}'_k(m_1)\rangle = 0] \\
&+ \frac{1}{2}\Pr[\mathcal{A}^{hy}\langle \text{Enc}_{pk}(k), \text{Enc}'_k(m_1)\rangle = 1]
\end{aligned} \tag{5.12}$$

Passo IV. Etapa final:

Somando as equações 5.6, 5.9 e 5.12) dos 3 passos anteriores e considerando que a soma de 3 funções desprezíveis é também desprezível, podemos definir uma função negl desprezível tal que:

$$\begin{aligned}
\frac{1}{2} + \text{negl}(n) &= \frac{1}{2} + (\text{negl}_1(n) + \text{negl}_2(n) + \text{negl}_3(n)) \\
&\geq \frac{1}{2}(\Pr[\mathcal{A}^{hy}\langle \text{Enc}_{pk}(k), \text{Enc}'_k(m_0)\rangle = 0] \\
&+ \Pr[\mathcal{A}^{hy}\langle \text{Enc}_{pk}(0^n), \text{Enc}'_k(m_0)\rangle = 1] \\
&+ \Pr[\mathcal{A}^{hy}\langle \text{Enc}_{pk}(0^n), \text{Enc}'_k(m_0)\rangle = 0] \\
&+ \Pr[\mathcal{A}^{hy}(\text{Enc}_{pk}(0^n), \text{Enc}'_k(m_1)\rangle = 1] \\
&+ \Pr[\mathcal{A}^{hy}\langle \text{Enc}_{pk}(0^n), \text{Enc}'_k(m_1)\rangle = 0] \\
&+ \Pr[\mathcal{A}^{hy}(\text{Enc}_{pk}(k), \text{Enc}'_k(m_1)\rangle = 1])
\end{aligned} \tag{5.13}$$

Observe que a complementaridade dos eventos nos permite concluir que:

$$\Pr[\mathcal{A}^{\mathrm{hy}}\langle \mathrm{Enc}_{\mathrm{pk}}(0^{\mathrm{n}}), \mathrm{Enc}'_{\mathrm{k}}(\mathrm{m}_0)\rangle = 1] + \Pr[\mathcal{A}^{\mathrm{hy}}\langle \mathrm{Enc}_{\mathrm{pk}}(0^{\mathrm{n}}), \mathrm{Enc}'_{\mathrm{k}}(\mathrm{m}_0)\rangle = 0] = 1 \tag{5.14}$$

$$\Pr[\mathcal{A}^{\mathrm{hy}}\langle \mathrm{Enc}_{\mathrm{pk}}(0^{\mathrm{n}}), \mathrm{Enc}'_{\mathrm{k}}(\mathrm{m}_1)\rangle = 1] + \Pr[\mathcal{A}^{\mathrm{hy}}\langle \mathrm{Enc}_{\mathrm{pk}}(0^{\mathrm{n}}), \mathrm{Enc}'_{\mathrm{k}}(\mathrm{m}_1)\rangle = 0] = 1 \tag{5.15}$$

Logo:

$$\begin{aligned} \frac{1}{2} + \mathrm{negl}(\mathrm{n}) \geq \frac{1}{2}(&\Pr[\mathcal{A}^{\mathrm{hy}}\langle \mathrm{Enc}_{\mathrm{pk}}(\mathrm{k}), \mathrm{Enc}'_{\mathrm{k}}(\mathrm{m}_0)\rangle = 0] \\ &+ \Pr[\mathcal{A}^{\mathrm{hy}}\langle \mathrm{Enc}_{\mathrm{pk}}(\mathrm{k}), \mathrm{Enc}'_{\mathrm{k}}(\mathrm{m}_1)\rangle) = 1]) \\ &= \Pr[\mathrm{PubK}^{\mathrm{cca}}_{\mathcal{A}^{\mathrm{hy}}\Pi^{\mathrm{hy}}(1^{\mathrm{n}})} = 1] \end{aligned} \tag{5.16}$$

Isso corresponde à equação 5.3, o que confirma que o sistema composto Π^{hy} é CCA-seguro.

Uma vez demonstrada a segurança do sistema principal, podemos analizar a segurança dos esquemas utilizados na construção dos índices. As demais cifras são estatísticamente idependentes da cifra principal. Isto é, utilizam chaves diferentes e cifram mensagens apenas remotamente relacionadas (representações numéricas de alguns elementos tomados do XML do arquétipo cifrado com o sistema principal). Além disso, não participam de nenhuma composição: não geram *input* uma para outra, nem para a cifra principal. Por isso, o leitor deverá encontrar a prova de segurança de cada sistema nos trabalhos originais (cf. referências a seguir).

O sistema de Paillier é não determinístico e tem a propriedade de indistinguibilidade semântica [89]. Isto é, a cifra não revela qualquer informação sobre o texto plano, mesmo quando o atacante consegue cifrar e decifrar algumas amostras. A hipótese de segurança desse sistema é a dificuldade do problema decisional da residuosidade composta, que, como demonstra o trabalho de Paillier, é tão intratável quanto o problema RSA. Por isso, esse sistema foi considerado seguro nessa análise. Note que, como discutido acima, as cifras geradas com esse sistema são independentes das demais, além de utilizar um par de chaves idenpendente. Portanto, esta cifra não influencia a segurança do sistema AES (que contém o registro completo), nem é influenciada pela presença das demais cifras.

O sistema de criptografia com preservação de ordem implementado neste trabalho é a parte mais delicada da segurança, pois tem uma taxa de vazamento de bits considerável. O sistema de Boldyreva *et al.* é determinístico, portanto não tem a propriedade de indistinguibilidade do texto cifrado. Tampouco pode atingir a indistinguibilidade semântica, uma vez que foi projetado para que a cifra revele alguma informação sobre o dado cifrado:

sua relação de ordem. Na verdade, como discutido no capítulo anterior, para qualquer sistema com preservação de ordem baseado em uma primitiva determinística, um atacante pode inferir até 50% dos bits do valor cifrado, a depender da quantidade e da dispersão das amostras que puder comparar [94].

Esse risco é mitigado no Safe-Record pelo fato de que o atacante nunca terá acesso a um número suficiente de amostras, pois cada prontuário é cifrado com uma chave diferente. Além disso, a dispersão dos dados nunca é tão grande: a maior parte dos dados (pressão arterial, peso, altura, idade, etc), é sempre observada dentro de intervalos curtos, numa mesma ordem de magnitude.

5.4 Detalhes da implementação

O CloudEHR, componente principal da solução, foi construído numa arquitetura cliente-servidor. O *back-end*, que roda na nuvem, foi escrito com o *framework* CodeIgniter, na linguagem PHP. Esse *framework* permite o desacoplamento entre as estruturas de dados, na camada de modelo, do banco de dados utilizado para persistência. Isto é, o mesmo código é plenamente funcional para todos os bancos suportados – entre eles, Oracle, PostgreSQL, MySQL e SQL Server.

O *front-end*, que roda no *browser* do usuário final, foi escrito em ECMAScript (JavaScript), com o uso do *framework* AngulaJS. O AngularJS é importante nessa solução por permitir a construção de uma aplicação cliente completa no modelo arquitetural MVC (Model-View-Controller) e, portanto, uma clara separação entre cliente e servidor. Na camada de modelo, temos as estruturas de dados e as respectivas regras de negócio. É também nessa camada que as classes fazem requisições à extensão BrowserTrust para cifrar e decifrar os dados à medida que são enviados ou recebidos da nuvem. Na camada de controle, temos o controle de fluxo (autenticação. autorização, etc) e da mensageria entre os diversos objetos. Na camada de apresentação, por fim, as classes responsáveis pela renderização de elementos da interface (formulários, gráficos, etc.) e por observar e reagir à interação do usuário.

No AngularJS, os modelos podem vincular suas estruturas de dados internas a serviços externos, de forma que uma alteração no dado se torna uma notificação automática ao serviço, e, uma mensagem recebida do serviço é também automaticamente revertida na atualização do dado, desde a camada de modelo até a de apresentação. É justamente nessa vinculação modelo/serviço que foram incorporados os gatilhos que disparam as requisições à extensão BrowserTrust para ciframento e deciframento dos dados.

A extensão também foi escrita em ECMAScript e usa extensivamente as APIs *runtime* e *extensions* do projeto Chromium, e, portanto, pode ser instaladas nos navegadores

Chromium, Google Chrome e Mozilla Firefox. São essas APIs que dão acesso a recursos internos do *browser*, tais como controle de sessão e análise de requisições de rede. As funcionalidades relacionadas à criptografia foram construídas sobre bibliotecas consolidadas e extensivamente testadas, tais como a SJCL (Stanford JavaScript Crypto Library) e CryptoJS, do Google. A extensão também traz uma implementação nova do criptosistema OPE de Boldyreva.

O aplicativo MobileID foi escrito com Ionic, um *framework* para construção de aplicativos HMA (*Hybrid Mobile App*) baseado no AngularJS. Aplicativos HMA são baseados em tecnologias HTML5 e rodam em *containers* especializados, de acordo com o sistema operacional (SO) do dispositivo. Esses *containers*, geralmente chamados WebViews ou UIWebViews, disponibilizam ao aplicativo tando as APIs do padrão HTML5 quanto APIs específicas do SO para acesso a recursos do dispositivo, incluindo o controle de periféricos.

A escolha por uma arquitetura HMA permitiu a harmonização das bibliotecas criptográficas: tanto o dispositivo principal quando o dispositivo móvel utilizam as mesmas bibliotecas. O aplicativo foi compilado (empacotado) com o uso da plataforma Apache Cordova, que produz instaláveis para uma grande variedade de dispositivos e sistemas operacionais. Foram testados instaláveis para Android e iOS.

A aplicação principal foi implantada no serviço de PaaS da Heroku, uma subsidiária da Salesforce. A aplicação persiste os dados através do *plug-in* ClearDB – um serviço especializado de Database-as-a-Service. A escolha por esse serviço se deu tanto por sua integração simples com o Heroku e também pela possibilidade de usar ferramentas genéricas de administração (Oracle Workbench, phpMyAdmin, etc.). Esse serviço também permitiu que a a aplicação fosse configurada para usar conexões TLS para comunicação com o banco de dados. Com algumas mudanças na configuração da aplicação e na camada de persistência, também foi possível implantar a aplicação no serviço AppEngine, a plataforma PaaS do Google.

5.4.1 Viabilidade técnica

Os testes realizados com o Safe-Record mostram que a solução proposta é tecnicamente viável e atende adequadamente aos requisitos de uma aplicação real de registro eletrônico em saúde. As limitações impostas pelos aspectos de segurança do projeto não impactam significativamente o desempenho da aplicação nem sua usabilidade.

Considerando-se o fluxo de trabalho em aplicações de RES comuns no mercado, talvez a alteração mais distintiva seja a introdução de uma política de autenticação por dois fatores. Mas essa técnica não tem relação direta com o ambiente de computação em nuvem, aplicando-se à segurança de qualquer aplicação e a qualquer modalidade de implantação. Considerando-se a arquitetura da aplicação, no entanto, o aspecto mais expressivo

é a adoção de vários esquemas criptográficos para o armazenamento e manipulação dos dados na nuvem.

Do ponto de vista do paciente, esse processo de encriptação é transparente. E este se beneficia com grandes ganhos de segurança e privacidade, sem ceder muito, ou praticamente nada, em sua experiência de uso do aplicativo. O clínico também não é afetado no uso comum do aplicativo, pois não notará atraso na leitura ou escrita dos registros clínicos. É certo que muitas operações criptográficas são realizadas no dispositivo do usuário, mas os cálculos mais pesados, que são utilizados para busca, comparação ou soma de dados encriptados, são executados na nuvem.

Do ponto de vista do pesquisador em saúde, que necessita realizar computações complexas sobre os dados, haverá alguma perda de velocidade nas respostas. Vale lembrar que os esquemas criptográficos usados permitem a realização de um número bastante limitado de operações aritméticas na nuvem, além da comparação de ordem de entradas de dados numéricos. Essa restrição é aceitável, em face dos ganhos de segurança.

5.4.2 Viabilidade econômica

Embora a arquitetura do Safe-Record não tenha tanto impacto em seu desempenho, certamente afeta o seu custo de operação. Em lugar de armazenar as entradas de um arquétipo em claro, é armazenada uma cifra simétrica, combinada a vários índices gerados com os sitemas OPE e AHE empregados (usados para a execução das operações de comparação ou soma na nuvem). Isso implica em maior demanda no canal de transporte e maior carga no serviço de armazenamento. Afetando, portanto, o custo dos serviços de nuvem, medidos em termos de espaço em disco ou volume de tráfego de entrada ou saída.

Dessa forma, quando confrontado com uma solução que simplesmente grave uma cópia (em claro) dos registros, o Safe-record certamente representa um custo operacional maior. Essa diferença foi considerada aceitável, no entanto, em face dos ganhos em segurança e, consequentemente, flexibilidade na administração de sistemas de informações em saúde. As organizações depositárias de tais informações, certamente, terão ganhos com a possibilidade de adoção de um serviço regular de computação em nuvem.

6 Conclusões

Este trabalho demonstrou que é possível garantir um nível adequado de segurança dos dados e de privacidade dos usuários finais de uma aplicação implantada em um ambiente de computação em nuvem com o uso adequado de esquemas criptográficos e outros mecanismos de segurança.

A solução apresentada partiu de um levantamento ostensivo das principais ameaças, bem como das mais recentes e avançadas soluções apresentadas pela comunidade acadêmica na literatura de referência na área. Como demonstrado no Capítulo 2, o problema mais relevante se refere à dificuldade de garantir a privacidade do usuário final e a segurança da informação, caso os dados transitem, sejam processados ou armazenados em claro na nuvem. Esse problema foi enfrentado, neste trabalho, com o uso de uma combinação de esquemas criptográficos utilizados no controle de acesso aos dados e em sua manipulação na nuvem.

O Capítulo 2 também apresenta as ameças mais pervasivas, atingindo todos os tipos de serviço da nuvem, que são fragilidades comuns às aplicações *web*. Essas fragilidades foram enfrentadas nesse trabalho, entre outros meios, com o emprego de uma técnica de autenticação por dois fatores e com a introdução de um *plug-in* de *browser* como elemento confiável.

Além disso, o aplicativo construído como prova de conceito demonstrou que essas propriedades de segurança podem ser aliadas à utilização de um padrão de modelagem de dados e de interoperabilidade funcional e semântica que se adéqua aos requisitos de negócio de uma aplicação real de registros eletrônicos em saúde. O OpenEHR, apresentado no Capítulo 3, foi o padrão utilizado.

Esse trabalho também demonstrou, no Capítulo 4, que o esquema aditivamente homomórfico de Paillier pode ser usado para a soma valores encriptados na nuvem. O resultado encriptado é entregue ao dispositivo do usuário, onde ocorre a decriptação e exibição do dado apenas se o usuário for devidamente autorizado (possuir a chave privada correspondente). Também foi apresentado o sistema de Boldyreva, um esquema com preservação de ordem, permite a realização de buscas de valores que atendam a dados intervalos, sem dar a conhecer o valor do dado encriptado nem o intervalo pesquisado.

6.1 Resultados obtidos

O quinto capítulo, por fim, demonstrou os detalhes de projeto, a arquitetura, a interação entre os diversos componentes e o potencial do Safe-Record, a aplicação construída como ferramenta de experimentação, como demonstração de solução de segurança útil para sistemas RES. O três produtos que demonstram a solução proposta neste trabalho foram disponibilizados na forma de software livre e aberto, nos seguintes repositórios do GitHub:

- CloudEHR: o aplicativo *web* que implementa os padrões OpenEHR para registros eletrônicos em saúde, em `https://github.com/safe-record/CloudEHR`;
- MobileID: o aplicativo para dispositivos móveis, utilizado no processo de autenticação por dois fatores, em `https://github.com/safe-record/MobileId`;
- BrowserTrust: o *plug-in* para *browsers* baseados no WebKit, utilizado para disponibilizar primitivas criptográficas (especialmente para implementar os criptossistemas de Paillier e Boldyreva) no navegador, em `https://github.com/safe-record/BrowserTrust`.

6.2 Indicações para trabalhos futuros

Os produtos desenvolvidos nesse trabalho podem ser aperfeiçoados e estendidos, contribuindo para a evolução da solução de segurança proposta. Do ponto de vista do componente semântico, específico para aplicações do tipo RES, a principal melhoria a ser realizada é a implementação de mais arquétipos ou até mesmo o desenvolvimento de uma solução mais genérica, que, a partir da definição de um arquétipo descrito em ADL possa gerar as representações AOM internas e executar *queries AQL*, sem perda nos requisitos de segurança e privacidade. Isso é importante para refinar e aperfeiçoar a solução proposta à medida que a complexidade do Safe-Record se aproxime da complexidade de uma aplicação real de RES, com conteúdo clínico mais diversificado.

Também é possível atender aos mesmos requisitos explorados nesta versão do Safe-Record utilizando outros esquemas criptográficos com características semelhantes aos selecionados. Seria interessante, por exemplo, testar o esquema de Benaloh [44] ou o de Melchor *et al* [82] para a cifra aditivamente homomórfica. Os esquemas propostos em [94] e [79] podem substituir a cifra com preservação de ordem. Também é possível explorar criptossistemas com outros homomorfismos, tais como a propriedade de homomorfismo multiplicativo do RSA original. A comparação com outras combinações de esquemas pode levantar informação relevante quanto às funcionalidades, o desempenho e custo de operação da aplicação.

Sob uma perspectiva mais geral, aplicável a qualquer aplicação na nuvem, também seria útil introduzir outros elementos de segurança, a fim de analisar suas contribuições, contrapondo-as aos respectivos impactos no desempenho e na viabilidade da solução. Uma sugestão seria um mecanismo de ofuscação da busca que realize operações fictícias (*dummy searches*), de forma a dificultar ao provedor, ou um atacante com acesso ao canal de comunicação, a identificação do real dado de interesse. Esse tipo de mecanismo também poderia dificultar a execução de ataques do tipo *timming side-channel*.

6.3 Publicações relacionadas a este trabalho

No decurso do trabalho de pesquisa exposto nesta dissertação, foram produzidos os seguintes trabalhos:

Publicados:

GONCALVES, R. Leonova, E. Puttini, R. Nascimento, A. A privacy-ensuring scheme for health data outsourcing. In *Cloud Technologies and Applications (Cloud- Tech), 2015 International Conference on* , pp.1–7, 6/2015.

SOUZA, S., Puttini, R. Security assessment for cloud applications. In *International Journal of Engineering Research and Technology.* 5(6), 6/2016.

Aceitos:

SOUZA, S. Gonćalves, R. Leonova, E. Puttini, R. Nascimento, A. Privacy-ensuring eletronic health records on the cloud. In *Concurrency and Computation: Practice and Experience. Special issue "Cloud Computing and Big Data: Technologies and Applications"*, (9/2016).

SOUZA, S. R, Puttini. Client-side encryption for privacy-sensitive applications on the cloud. In The HOLACONF - Cloud Forward: From Distributed to Complete Computing, Proceedings of. Matrid, 10/2016.

Referências

[1] Devdatta Akhawe, Frederik Braun, Francois Marier, e Joel Weinberger. Subresource integrity, 11 2015. 86, 87

[2] Nadhem J. AlFardan e Kenneth G. Paterson. Lucky thirteen: Breaking the tls and dtls record protocols. In *2013 IEEE Symposium on Security and Privacy*. IEEE, 2013. 22

[3] Aaron Alva, Olivier Caleff, Greg Elkins, Allen Lum, Keith Pasley, Satheesh Sudarsan, Vinoth Sivasubramanian, e Rajeev Venkitaraman. The notorious nine: Cloud computing top threats in 2013, 2013. 1

[4] Dino Macedo Amaral. *Hy-SAIL: Uma nova abordagem para distribuićã0 e armazenamento de informaćões em ambientes de computaćão em nuvem*. Tese (Doutorado), PPGEE, 2013. 2

[5] Ross Anderson. *Security Engineering*. Winley, Cambridge, 2001. 16

[6] Richard Horne Andrew Miller e Chris Potter. 2015 information security breaches survey. Technical report, 2015. 2

[7] Claudio A. Ardagna, Sabrina De Capitani Di Vimercati, Sara Foresti, Tyrone W. Grandison, Sushil Jajodia, e Pierangela Samarati. Access control for smarter healthcare using policy spaces. *Computer Security*, 29(8):848–858, Nov 2010. 77

[8] Michael Armbrust, Armando Fox, Rean Griffith, Anthony D. Joseph, Randy Katz, Andy Konwinski, Gunho Lee, David Patterson, Ariel Rabkin, Ion Stoica, e Matei Zaharia. A view of cloud computing. *Communications of the ACM*, 53(4):50–58, 4 2010. 15

[9] Koray Atalag, Hong Yul Yang, Ewan Tempero, e James R Warren. Evaluation of software maintainability with open ehr - a comparison of architectures. *International journal of medical informatics*, 83(11):849–859, November 2014. 9, 33

[10] Adam Barker, Blesson Varghese, Jonathan Stuart Ward, e Ian Sommerville. Academic cloud computing research: Five pitfalls and five opportunities. In *6th USENIX Workshop on Hot Topics in Cloud Computing (HotCloud 14)*, Philadelphia, PA, June 2014. USENIX Association. 2

[11] Thomas Beale. From semantics to services - the openehr experience. In *OMG SOA Conference in Healthcare*, Washington, DC, 2011. 38

[12] Mihir Bellare e Phillip Rogaway. Optimal asymmetric encryption - how to encrypt with rsa. In *Lecture Notes in Computer Science*, volume 950, pages 92–111. Springer-Verlag, 1995. 57, 85, 95

[13] Mihir Bellare e Phillip Rogaway. Pss: Provably secure encoding method for digital signatures, 1998. 88

[14] Josh Benaloh, Melissa Chase, Eric Horvitz, e Kristin Lauter. Patient controlled encryption: Ensuring privacy of electronic medical records. In *CCSW'09*, Nov 2009. 4, 30, 31, 72

[15] I. Berges, J. Bermudez, e A. Illarramendi. Toward semantic interoperability of electronic health records. *IEEE Transactions on Information Technology in Biomedicine*, 16(3):424–431, May 2012. 9, 29, 30, 33

[16] Sören Bleikertz, Matthias Schunter, Christian W. Probst, Dimitrios Pendarakis, e Konrad Eriksson. Security audits of multi-tier virtual infrastructures in public infrastructure clouds. In *CCSW'10*, 2010. 24

[17] Reza Curtmola Bo Chen. Poster: Robust dynamic remote data checking for public clouds. In *CCS'12*. ACM, ACM Press, 2012. 26

[18] Alexandra Boldyreva, Nathan Chenette, Younho Lee, e Adam O'Neill. Order-preserving symmetric encryption. In *Proceedings of the 28th Annual International Conference on Advances in Cryptology: The Theory and Applications of Cryptographic Techniques*, EUROCRYPT '09, pages 224–241, Berlin, Heidelberg, 2009. Springer-Verlag. 14, 66

[19] Alexandra Boldyreva, Nathan Chenette, e Adam O'Neill. Order-preserving encryption revisited: Improved security analysis and alternative solutions. In *Proceedings of the 31st Annual Conference on Advances in Cryptology*, CRYPTO'11, pages 578–595, Berlin, Heidelberg, 2011. Springer-Verlag. 67

[20] Joppe W. Bos, Kristin Lauter, e Michael Naehrig. Private predictive analysis on encrypted medical data. Cryptology ePrint Archive, Report 2014/336, 2014. 1, 32, 72

[21] BRASIL. Estratégia do registro eletrônico de saúde, 2016. 5, 9

[22] Jon-Michael Brook, Scott Field, Dave Shackleford, Vic Hargrave, Laurie Jameson, e Michael Roza. The treacherour 12: Cloud computing top threats in 2016, 2016. 1

[23] Daniel R. L. Brown. What hashes make rsa-oaep secure? Cryptology ePrint Archive, Report 2006/223, 2006. 85, 95

[24] Shakeel Butt, H. Andres Lagar-Cavilla, Abhinav Srivastava, e Vinod Ganapathy. Self-service cloud computing. In *CCS'12*, pages 253–264. ACM Press, October 2012. CloudVisor, XenBlanket - VM nesting, protects agains dom0, side chanells. 2, 24

[25] Jeffrey C. Carver, Morgan Burcham, Sedef Akinli Kocak, Ayse Bener, Michael Felderer, Matthias Gander, Jason King, Jouni Markkula, Markku Oivo, Clemens Sauerwein, e Laurie Williams. Establishing a baseline for measuring advancement in the science of security: An analysis of the 2015 ieee security & privacy proceedings. In *Proceedings of the Symposium and Bootcamp on the Science of Security*, HotSos '16, pages 38–51, New York, NY, USA, 2016. ACM. 2

[26] H.M. Chao, S.H. Twu, e C.M. Hsu. A secure identification access control scheme for accessing healthcare information systems. In *Information Technology Applications in Biomedicine, 2003. 4th International IEEE EMBS Special Topic Conference on*, pages 122–125, April 2003. 72, 82

[27] Melissa Chase e Kristin Lauter. An anonymous health care system. In *USENIX Security Symposium*, 2010. 4, 32

[28] Jeremy Clark e Paul C. van Oorschot. Sok: Ssl and https: Revisiting past challenges and evaluating certificate trust model enhancements. In *2013 IEEE Symposium on Security and Privacy*. IEEE, 2013. 22

[29] Wendy L. Currie e Jonathan J. M. Seddon. A cross-country study of cloud computing policy and regulation in healthcare. In *22st European Conference on Information Systems, ECIS 2014, Tel Aviv, Israel, June 9-11, 2014*, 2014. 52

[30] Alexei Czeskis, Michael Dietz, Tadayoshi Kohno, Dan Wallach, e Dirk Balfanz. Strengthening user authentication through opportunistic cryptographic identity assertions. In *CCS'12*. ACM Press, 2012. 21

[31] Joan Daemen e Vincent Rijmen. Aes proposal: Rijndael, 1999. 59

[32] Sociedade Brasileira de Informática em Saúde. *Manual de Certificação para Sistemas de Registro Eletrônico em Saúde v4.2*, 2016. 6, 29

[33] Flávio de Oliveira Silva. Controle de acesso a web services baseado em um protocolo de autenticação segura. Dissertação (Mestrado), Universidade Federal de Uberândia, 2004. 21, 53

[34] Whitfield Diffie e Martin Hellman. Privacy and authentication: An introduction to cryptography. volume 67, pages 397–427. March 1979. 60, 94

[35] Shenlu Wang Dongxi Liu. Demo: Query encrypted databases practically. In *CCS'12*, ACM 978-1-4503-1651-4/12/10. ACM, ACM Press, 2012. 25, 72

[36] Adam Doupé, Weidong Cui, Mariusz H. Jakubowski, Marcus Peinado, Christopher Kruegel, e Giovanni Vigna. dedacota: toward preventing server-side xss via automatic code and data separation. In *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, CCS '13, pages 1205–1216, New York, NY, USA, 2013. ACM. 25

[37] C. Chris Erway, Alptekin Küpçü, Charalampos Papamanthou, e Roberto Tamassia. Dynamic provable data possession. *ACM Trans. Inf. Syst. Secur.*, 17(4):15:1–15:29, April 2015. 2

[38] R. J. Richardson et al. *Pesquisa social: métodos e técnicas*. Atlas, São Paulo, 1999. 7

[39] Simone Fabiano Mendes et al. Uma análise da implantação do padrão de troca de informação em saúde suplementar no brasil. *Journal of Health Informatics*, 1(2), 2009. 4

[40] Sascha Fahl, Marian Harbach, Thomas Muders, Matthew Smith, Lars Baumgartner, e Bernd Freisleben. Why eve and mallory love android: An analysis of android ssl (in)security. In *CCS'12*, pages 50–61. ACM Press, October 2012. 22

[41] The OpenEHR Foundation. *OpenEHR Clinical knowledge Manager*, 2010. 9, 10, 34

[42] The OpenEHR Foundation. *OpenEHR Archetype Query Language*, 1.0.3 edition, 2015. 13

[43] The OpenEHR Foundation. *OpenEHR DataType Information Model*, 1.0.2 edition, 2015. x, 38, 42

[44] Laurent Fousse, Pascal Lafourcade, e Mohamed Alnuaimi. Benaloh's dense probabilistic encryption revisited. *CoRR*, abs/1008.2991, 2010. 64, 103

[45] Eiichiro Fujisaki, Tatsuaki Okamoto, David Pointcheval, e Jacques Stern. *Advances in Cryptology — CRYPTO 2001: 21st Annual International Cryptology Conference, Santa Barbara, California, USA, August 19–23, 2001 Proceedings*, chapter RSA-OAEP Is Secure under the RSA Assumption, pages 260–274. Springer Berlin Heidelberg, Berlin, Heidelberg, 2001. 85, 95

[46] Sebastian Garde, Evelyn Hovenga, Jasmin Buck, e Petra Knaup. Expressing clinical data sets with openehr archetypes: A solid basis for ubiquitous computing. *International Journal of Medical Informatics*, 76(3):334–341, 2015. 30, 33, 40

[47] Dimitris Geneiatakis, Georgios Portokalidis, Vasileios P. Kemerlis, e Angelos D. Keromytis. Adaptive defenses for commodity software through virtual application partitioning. In *CCS'12*, pages 133–144. ACM Press, October 2012. 2, 25, 74

[48] Craig Gentry. *A fully homomorphic encryption scheme*. Tese (Doutorado), Stanford University, 2009. `crypto.stanford.edu/craig`. 62

[49] Jack D. Glazier e Ronald R. POWELL. Qualitative research in information management, 1992. 8

[50] R. Goncalves, E. Leonova, R. Puttini, e A. Nascimento. A privacy-ensuring scheme for health data outsourcing. In *Cloud Technologies and Applications (CloudTech), 2015 International Conference on*, pages 1–7, June 2015. x, 4, 29, 31, 72, 74

[51] Shay Gueron e Yehuda Lindell. Gcm-siv: Full nonce misuse-resistant authenticated encryption at under one cycle per byte. In *Proceedings of the 22Nd ACM SIGSAC Conference on Computer and Communications Security*, CCS '15, pages 109–119, New York, NY, USA, 2015. ACM. 62

[52] Joaquín Guillén, Javier Miranda, Juan Manuel Murillo, e Carlos Canal. Developing migratable multicloud applications based on mde and adaptation techniques. In *Proceedings of the Second Nordic Symposium on Cloud Computing & Internet Technologies*, NordiCloud '13, pages 30–37, New York, NY, USA, 2013. ACM. 26

[53] Pablo Pazos Gutiérrez. Towards the implementation of an openehr-based open source ehr platform (a vision paper). *Studies in health technology and informatics*, 216:45–49, 2015. 9, 30, 33

[54] Stuart Haber e Benny Pinkas. Securely combining public-key cryptosystems. In *Proceedings of the 8th ACM Conference on Computer and Communications Security*, CCS '01, pages 215–224, New York, NY, USA, 2001. ACM. 88

[55] Shai Halevi, Danny Harnik, Benny Pinkas, e Alexandra Shulman-Peleg. Proofs of ownership in remote storage systems. In *Proceedings of the 18th ACM Conference on Computer and Communications Security*, CCS '11, pages 491–500, New York, NY, USA, 2011. ACM. 26

[56] Dennis Hofheinz, John Malone-Lee, e Martijn Stam. Obfuscation for cryptographic purposes. Cryptology ePrint Archive, Report 2006/463, 2006. `http://eprint.iacr.org/`. 70

[57] Eman Hossny, Sherif Khattab, Fatma A. Omara, e Hesham Hassan. Semantic-based generation of generic-api adapters for portable cloud applications. In *Proceedings of the 3rd Workshop on CrossCloud Infrastructures & Platforms*, CrossCloud '16, pages 1:1–1:5, New York, NY, USA, 2016. ACM. 27

[58] Jiankun Hu, Hsiao-Hwa Chen, e Ting-Wei Hou. A Hybrid Public Key Infrastructure Solution (HPKI) for HIPAA Privacy/Security Regulations. *Comput. Stand. Interfaces*, 32(5-6):274–280, Oct 2010. 52

[59] M. Iorga e A. Karmel. Managing risk in a cloud ecosystem. *IEEE Cloud Computing*, 2(6):51–57, 11 2015. 1, 20

[60] Information technology – General-Purpose Datatypes (GPD). Standard, International Organization for Standardization, Geneva, CH, December 2007. 44

[61] Health informatics – Electronic health record communication – Part 1: Reference model. Standard, International Organization for Standardization, Geneva, CH, February 2008. 33

[62] Information technology – Security techniques – Encryption algorithms – Part 3: Block ciphers. Standard, International Organization for Standardization, Geneva, CH, December 2010. 59

[63] Data elements and interchange formats – Information interchange – Representation of dates and times. Standard, International Organization for Standardization, Geneva, CH, December 2004. 44

[64] Wayne Jansen e Timothy Grance. *Guidelines on Security and Privacy in Public Cloud Computing*, 2011. 16

[65] Jing Jin, Gail-Joon Ahn, Hongxin Hu, Michael J. Covington, e Xinwen Zhang. Patient-centric authorization framework for electronic healthcare services. *Computers & Security*, 30(23):116 – 127, 2011. Special Issue on Access Control Methods and Technologies. 12, 30, 72

[66] Aaron Johnson, Chris Wacek, Rob Jansen, Micah Sherr, e Paul Syverson. Users get routed: Traffic correlation on tor by realistic adversaries. In *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security*, CCS '13, pages 337–348, New York, NY, USA, 2013. ACM. 23

[67] Jakob Jonsson e Burt Kaliski. Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1. RFC 3447, Internet Engineering Task Force, February 2003. 57, 58

[68] Burt Kaliski. PKCS #5: Password-Based Cryptography Specification Version 2.0. RFC 2898, Internet Engineering Task Force, September 2000. 55, 79

[69] Bonnie Kaplan e Dennis Duchon. Combining qualitative and quantitative methods in information systems research: A case study. *MIS Q.*, 12(4):571–586, 12 1988. 7

[70] Florian Kerschbaum. Client-controlled cloud encryption. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, CCS '14, pages 1542–1543, New York, NY, USA, 2014. ACM. 2, 71

[71] Leonard Kleinrock. A vision for the internet. *ST Journal for Research*, 2(1):4–5, novembro 2005. 15

[72] Rashmi Kumari, Mohsen Alimomeni, e Reihaneh Safavi-Naini. Performance analysis of linux rng in virtualized environments. In *Proceedings of the 2015 ACM Workshop on Cloud Computing Security Workshop*, CCSW '15, pages 29–39, New York, NY, USA, 2015. ACM. 23

[73] Heather Leslie. International developments in openehr archetypes and templates. *Health Information Management Journal*, 37(1), 2008. 9, 33, 38

[74] Changbin Liu e Yun Mao. Inception: Towards a nested cloud architecture. In *Proceedings of the 5th USENIX Workshop on Hot Topics in Cloud Computing - HotCloud'13*. USENIX, 2013. 2, 24

[75] Adriana Lopez-Alt, Eran Tromer, e Vinod Vaikuntanathan. On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption. Cryptology ePrint Archive, Report 2013/094, 2013. 63

[76] Kacha Lynda, Oukid-Khouas Saliha, e Benblidia Nadjia. Data security and privacy in e-health cloud: Comparative study. In *Proceedings of the International Conference on Intelligent Information Processing, Security and Advanced Communication*, IPAC '15, pages 8:1–8:6, New York, NY, USA, 2015. ACM. 1

[77] Hans Löhr, Ahmad-Reza Sadeghi, e Marcel Winandy. Securing the e-health cloud. In *Proceedings of the 1st ACM International Health Informatics Symposium*, IHI '10, pages 220–229, New York, NY, USA, 2010. ACM. 31, 82

[78] Aanchal Malhotra, Isaac E. Cohen, Erik Brakke, e Sharon Goldberg. Attacking the network time protocol. 2015. 22

[79] Tal Malkin, Isamu Teranishi, e Moti Yung. Order-preserving encryption secure beyond one-wayness. Cryptology ePrint Archive, Report 2013/409, 2013. 66, 103

[80] Kenneth D. Mandl, William W. Simons, William C. R. Crawford, e Jonathan M. Abbett. Indivo: a personally controlled health record for health information exchange and communication. In *BMC Medical Informatics and Decision Making*, number (7)25, 2007. 31

[81] Andreas Mayer, Marcus Niemietz, Vladislav Mladenov, e Jörg Schwenk. Guardians of the clouds: When identity providers fail. In *Proceedings of the 6th Edition of the ACM Workshop on Cloud Computing Security*, CCSW '14, pages 105–116, New York, NY, USA, 2014. ACM. 82

[82] Carlos Aguilar Melchor, Philippe Gaborit, e Javier Herranz. Additively homomorphic encryption with d-operand multiplications. Cryptology ePrint Archive, Report 2008/378, 2008. `http://eprint.iacr.org/`. 64, 103

[83] Rob Meredith. Electronic health records: Achieving an effective and ethical legal and recordkeeping framework, 2004. 47

[84] Travis Muirhead. Object databases and object persistence framework for openehr. Dissertação (Mestrado), University of South Australia, 2014. 47

[85] Shivaramakrishnan Narayan, Martin Gagné, e Reihaneh Safavi-Naini. Privacy preserving ehr system using attribute-based infrastructure. ACM, October 2010. 12, 72

[86] Nick Nikiforakis, Alexandros Kapravelos, Wouter Joosen, Christopher Kruegel, Frank Piessens, e Giovanni Vigna. Cookieless monster: Exploring the ecosystem of web-based device fingerprinting. In *2013 IEEE Symposium on Security and Privacy*. IEEE Computer Society, IEEE, 2013. 23

[87] Valeria Nikolaenko, Udi Weinsberg, Stratis Ioannidis, Marc Joyeand Dan Boneh, e Nina Taft? Privacy-preserving ridge regression on hundreds of millions of records. In *2013 IEEE Symposium on Security and Privacy*. IEEE, 2013. 1, 25, 66

[88] NIST-ITL. Specification for the advanced encryption standard (aes). Federal Information Processing Standards Publication 197, 2001. 59, 85

[89] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *IN ADVANCES IN CRYPTOLOGY - EUROCRYPT 1999*, pages 223–238. Springer-Verlag, 1999. 14, 64, 98

[90] Rupa Parameswaran. *A Robust Data Obfuscation Approach for Privacy Preserving Collaborative Filtering*. Tese (Doutorado), Atlanta, GA, USA, 2006. AAI3233574. 70

[91] Douglas F. Parkhill. *The challenge of the computer utility*. Addison-Wesley Professional, 1966. 15

[92] Peter Mell and Timothy Grance. *The NIST Definition of Cloud Computing*, 2011. 1, 15

[93] Daryl Plummer. The business landscape of cloud computing, 2012. 15

[94] Raluca Ada Popa, Frank H. Li, e Nickolai Zeldovich. An ideal-security protocol for order-preserving encoding. Cryptology ePrint Archive, Report 2013/129, 2013. `http://eprint.iacr.org/`. 66, 69, 99, 103

[95] Raluca Ada Popa, Emily Stark, Jonas Helfer, Steven Valdez, Nickolai Zeldovich, M. Frans Kaashoek, e Hari Balakrishnan. Building web applications on top of encrypted data using mylar. *11th USENIX Symposium on Networked Systems Design and Implementation*, 2014. 12, 25, 32, 72

[96] Raluca Ada Popa e Nickolai Zeldovich. Multi-key searchable encryption. Cryptology ePrint Archive, Report 2013/508, 2013. 25, 32

[97] Niels Provos e David Mazières. A future-adaptable password scheme. 1999. 53

[98] James Randall, Burt Kaliski, John Brainard, e Sean Turner. Use of the RSA-KEM Key Transport Algorithm in the Cryptographic Message Syntax (CMS). Proposed Standard 5990, Internet Engineering Task Force, September 2010. 57

[99] R[onald] L. Rivest, A[di] Shamir, e L[eonard M.] Adleman. A method for obtaining digital signatures and public-key cryptosystems. *CACM*, 26(1):96–99. 56

[100] C. Selltiz. *Métodos de pesquisa nas relações sociais*. Herder, São Paulo, 1967. 9

[101] Victor Shoup. Oaep reconsidered. *J. Cryptol.*, 15(4):223–249, September 2002. 90, 95

[102] George Silowash. Common sense guide to mitigating the insider threat v4, 2012. 24

[103] Duygu Sinanc e Seref Sagiroglu. A review on cloud security. In *Proceedings of the 6th International Conference on Security of Information and Networks*, SIN '13, pages 321–325, New York, NY, USA, 2013. ACM. 2

[104] Juraj Somorovsky, Mario Heiderich, Meiko Jensen, J́org Schwenk, Nils Gruschka, e Luigi Lo Iacono. All your clouds are belong to us - security analysis of cloud management interfaces. In *CCSW'11*, 2011. 21

[105] Stefano M P C Souza e Ricardo S Puttini. Security assessment for cloud applications. 5, 6 2016. 18

[106] Emil Stefanov e Elaine Shi. Multi-cloud oblivious storage. In *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security*, CCS '13, pages 247–258, New York, NY, USA, 2013. ACM. 26

[107] Emil Stefanov e Elaine Shi. Oblivistore: High performance oblivious cloud storage. In *2013 IEEE Symposium on Security and Privacy*. IEEE Computer Science, 2013. 2, 26

[108] A. Sunyaev, D. Chornyi, C. Mauro, e H. Krcmar. Evaluation framework for personal health records: Microsoft healthvault vs. google health. In *System Sciences (HICSS), 2010 43rd Hawaii International Conference on*, pages 1–10, Jan 2010. 31

[109] Peter Szolovits, Jon Doyle, William J. Long, Isaac Kohane, e Stephen G. Pauker. Guardian angel: Patient-centered health information systems. Technical report, CSAIL - MIT, Cambridge, MA, USA, 1994. 30

[110] Mohammad Mahdi Tajiki e Mohammad Ali Akhaee. Secure and privacy preserving keyword searching cryptography. In *Information Security and Cryptology (ISCISC), 2014 11th International ISC Conference on*, pages 226–230, Sept 2014. 25

[111] David Taylor, Tom Wu, Nikos Mavrogiannopoulos, e Trevor Perrin. Using the Secure Remote Password (SRP) protocol for TLS Authentication. RFC 5054, Internet Engineering Task Force, November 2007. 54, 79

[112] Jin Tong, Jian Mao, Robert Bohn, John Messina, Lee Badger, e Dawn Leaf. *NIST Cloud Computing Reference Architecture*, 2011. 15, 18

[113] Shruti Tople, Shweta Shinde, Zhaofeng Chen, e Prateek Saxena. Autocrypt: Enabling homomorphic computation on servers to protect sensitive web content. In *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security*, CCS '13, pages 1297–1310, New York, NY, USA, 2013. ACM. 24, 71

[114] Sean Turner. Asymmetric Key Packages. Proposed Standard 5958, Internet Engineering Task Force, August 2010. 79

[115] Marten van Dijk, Ari Juels, e Alina Oprea. Hourglass schemes: How to prove that cloud files are encrypted. In *CCS'12*. ACM Press, 2012. 26

[116] Ceusters W e Smith B. Strategies for referent tracking in electronic health. *Journal of Biomedical Informatics*, 39(3):288–298, 2006. 42, 72

[117] Mark Weiser. The computer for the 21st century. *SIGMOBILE Mob. Comput. Commun. Rev.*, 3(3):3–11, July 1999. 71

[118] Mike West. Content security policy level 3, 1 2016. 86

[119] Duane C. Wilson e Giuseppe Ateniese. To Share or Not to Share in client-side encrypted clouds. volume 8783. Elsevier, 2014. 2, 25, 71, 78

[120] Thomas Wu. The secure remote password protocol. In *In Proceedings of the 1998 Internet Society Network and Distributed System Security Symposium*, pages 97–111, 1998. 54

[121] Thomas Wu. Srp-6: Improvements and refinements to the the secure remote password protocol, October 2002. 54, 86

[122] Liang Xiao, Bo Hu, Madalina Croitoru, Paul Lewis, e Srinandan Dasmahapatra. A knowledgeable security model for distributed health information systems. *Computers & Security*, 29(3):331 – 349, 2010. Special issue on software engineering for secure systems. 73

[123] R. K. Yin. *Case Study Research: Design and Methods*. Beverly Hills, 1984. 8

[124] Yihua Zhang e Marina Blanton. Efficient dynamic provable possession of remote data via balanced update trees. In *Proceedings of the 8th ACM SIGSAC Symposium on Information, Computer and Communications Security*, ASIA CCS '13, pages 183–194, New York, NY, USA, 2013. ACM. 2

A *Template* Operacional RAS-AB

```
<?xml version="1.0"?>
<template xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns="openEHR/v1/Template">
  <id>399c81e4-0998-42fd-832e-5673f9798798</id>
  <name>RAS-AB</name>
  <description>
    <lifecycle_state>Initial</lifecycle_state>
    <details>
      <purpose>Resumo do registro de atendimento simplificado da aten\c{a}\~o b\'{a}sica</purpose>
      <use />
      <misuse />
    </details>
    <other_details>
      <item>
        <key>MetaDataSet:Sample Set </key>
        <value>Template metadata sample set </value>
      </item>
      <item>
        <key>Acknowledgements</key>
        <value />
      </item>
      <item>
        <key>Business Process Level</key>
        <value />
      </item>
      <item>
        <key>Care setting</key>
        <value />
      </item>
      <item>
        <key>Client group</key>
        <value />
      </item>
      <item>
        <key>Clinical Record Element</key>
        <value />
      </item>
      <item>
        <key>Copyright</key>
        <value />
      </item>
      <item>
        <key>Issues</key>
        <value />
      </item>
      <item>
        <key>Owner</key>
        <value />
      </item>
      <item>
        <key>Sign off</key>
        <value />
      </item>
      <item>
```

```
      <key>Speciality</key>
      <value />
    </item>
    <item>
      <key>User roles</key>
      <value />
    </item>
  </other_details>
</description>
<definition xsi:type="COMPOSITION" archetype_id="openEHR-EHR-COMPOSITION.encounter.v1"
concept_name="Encontro" name=" &quot;Registro de Atendimento Simplificado da Ateno Bsica -
RAS/AB&quot;">
  <Content xsi:type="ADMIN_ENTRY" archetype_id="openEHR-EHR-ADMIN_ENTRY.admission-sus.v1"
  concept_name="Registro de admisso na AB" path="/content">
    <Rule path="/data[at0001]/items[at0073]" hide_on_form="true" />
    <Rule path="/data[at0001]/items[at0073]/items[at0077]" max="0" />
    <Rule path="/data[at0001]/items[at0073]/items[at0078]" max="0" />
    <Rule path="/data[at0001]/items[at0073]/items[at0079]" max="0" />
    <Rule path="/data[at0001]/items[at0073]/items[at0104]" max="0" />
    <Rule path="/data[at0001]/items[at0073]/items[at0101]" max="0" />
    <Rule path="/data[at0001]/items[at0073]/items[at0102]" max="0" />
    <Rule path="/data[at0001]/items[at0073]/items[at0103]" max="0" />
    <Rule path="/data[at0001]/items[at0073]/items[at0084]" max="0" />
    <Rule path="/data[at0001]/items[at0073]/items[at0081]" max="0" />
    <Rule path="/data[at0001]/items[at0013]" max="0" />
    <Rule path="/data[at0001]/items[at0131]" max="0" />
    <Rule path="/data[at0001]/items[at0133]" max="0" />
    <Rule path="/data[at0001]/items[at0023]" max="0" />
    <Rule path="/data[at0001]/items[at0094]" max="0" />
    <Rule path="/data[at0001]/items[at0098]" max="0" />
    <Rule path="/data[at0001]/items[at0025]" max="0" />
    <Rule path="/data[at0001]/items[at0121]" max="0" />
    <Rule path="/data[at0001]/items[at0041]" max="0" />
    <Rule path="/data[at0001]/items[at0051]" max="0" />
    <Rule path="/data[at0001]/items[at0061]" max="0" />
    <Rule path="/data[at0001]/items[at0066]" max="0" />
    <Rule path="/data[at0001]/items[at0135]" max="0" />
    <Rule path="/data[at0001]/items[at0071]" hide_on_form="true" />
  </Content>
  <Content xsi:type="SECTION" archetype_id="openEHR-EHR-SECTION.adhoc_sus.v1" \\
  concept_name="Cabealho Adhoc" max="1" path="/content" name="Resumo do atendimento">
    <Item xsi:type="SECTION" archetype_id="openEHR-EHR-SECTION.adhoc_sus.v1"
    \\concept_name="Cabealho Adhoc" max="1" path="/items" name="Antropometria">
      <Item xsi:type="OBSERVATION" archetype_id="openEHR-EHR-OBSERVATION.body_weight.v1"
      \\concept_name="Peso corporal" path="/items">
        <Rule path="/data[at0002]/events[at0003]" hide_on_form="true" />
        <Rule path="/data[at0002]/events[at0003]/data[at0001]" hide_on_form="true" />
        <Rule path="/data[at0002]/events[at0003]/data[at0001]/items[at0024]" min="1" />
        <Rule path="/data[at0002]/events[at0003]/state[at0008]" hide_on_form="true" />
        <Rule path="/data[at0002]/events[at0003]/state[at0008]/items[at0009]" max="0" />
        <Rule path="/data[at0002]/events[at0003]/state[at0008]/items[at0025]" max="0" />
        <Rule path="/protocol[at0015]" hide_on_form="true" />
      </Item>
      <Item xsi:type="OBSERVATION" archetype_id="openEHR-EHR-OBSERVATION.height.v1"
      \\concept_name="Altura / comprimento" path="/items">
        <Rule path="/data[at0001]/events[at0002]" hide_on_form="true" />
        <Rule path="/data[at0001]/events[at0002]/data[at0003]" hide_on_form="true" />
        <Rule path="/data[at0001]/events[at0002]/data[at0003]/items[at0018]" max="0" />
        <Rule path="/data[at0001]/events[at0002]/state[at0013]" hide_on_form="true" />
        <Rule path="/data[at0001]/events[at0002]/state[at0013]/items[at0014]" max="0" />
        <Rule path="/data[at0001]/events[at0002]/state[at0013]/items[at0019]" max="0" />
        <Rule path="/protocol[at0007]" hide_on_form="true" />
      </Item>
    </Item>
    <Item xsi:type="EVALUATION" archetype_id="openEHR-EHR-EVALUATION.vaccination_summary-sus.v1"
```

```
120       \\concept_name="Sumrio do estado de imunizao" max="1" path="/items" name="Imunizao">
121         <Rule path="/data[at0001]/items[at0002]" max="0" />
122         <Rule path="/data[at0001]/items[at0003]" max="0" />
123         <Rule path="/data[at0001]/items[at0008]" max="0" />
124         <Rule path="/data[at0001]/items[at0009]" max="0" />
125       </Item>
126       <Item xsi:type="EVALUATION" archetype_id="openEHR-EHR-EVALUATION.aleitamento_materno.v1"
127       \\concept_name="Aleitamento materno" max="1" path="/items" name="Criana">
128         <Rule path="/data[at0001]/items[at0002]" max="0" />
129         <Rule path="/data[at0001]/items[at0003]/items[at0004]" max="0" />
130         <Rule path="/data[at0001]/items[at0003]/items[at0005]" max="0" />
131         <Rule path="/data[at0001]/items[at0003]/items[at0009]" max="0" />
132         <Rule path="/data[at0001]/items[at0007]" max="0" />
133         <Rule path="/data[at0001]/items[at0008]" max="0" />
134         <Rule path="/data[at0001]/items[at0010]" max="0" />
135       </Item>
136       <Item xsi:type="EVALUATION" archetype_id="openEHR-EHR-EVALUATION.pregnancy-sus.v1"
137       \\concept_name="Sumrio de gravidez" max="1" path="/items" name="Gestante">
138         <Rule path="/data[at0001]/items[at0002]" hide_on_form="true" />
139         <Rule path="/data[at0001]/items[at0007]" hide_on_form="true" />
140         <Rule path="/data[at0001]/items[at0007]/items[at0009]" max="0" />
141         <Rule path="/data[at0001]/items[at0018]" hide_on_form="true" />
142         <Rule path="/data[at0001]/items[at0018]/items[at0021]" max="0" />
143         <Rule path="/data[at0001]/items[at0018]/items[at0017]" max="0" />
144         <Rule path="/data[at0001]/items[at0018]/items[at0031]" max="0" />
145         <Rule path="/data[at0001]/items[at0018]/items[at0032]" max="0" />
146         <Rule path="/data[at0001]/items[at0019]/items[at0013]" max="0" />
147         <Rule path="/data[at0001]/items[at0019]/items[at0015]" max="0" />
148         <Rule path="/data[at0001]/items[at0019]/items[at0036]" max="0" />
149         <Rule path="/data[at0001]/items[at0019]/items[at0034]" max="0" />
150         <Rule path="/data[at0001]/items[at0019]/items[at0035]" max="0" />
151         <Rule path="/data[at0001]/items[at0019]/items[at0038]" max="0" />
152         <Rule path="/data[at0001]/items[at0019]/items[at0053]" max="0" />
153         <Rule path="/data[at0001]/items[at0108]" max="0" />
154         <Rule path="/data[at0001]/items[at0109]" max="0" />
155       </Item>
156       <Item xsi:type="EVALUATION"
157       archetype_id="openEHR-EHR-EVALUATION.modalidade_atencao_domiciliar.v1"
158       concept_name="Modalidade de atencao domiciliar" max="1" path="/items"
159       name="Ateno domiciliar" />
160       <Item xsi:type="SECTION" archetype_id="openEHR-EHR-SECTION.adhoc_sus.v1"
161       concept_name="Cabealho Adhoc" max="1" path="/items" name="Problemas">
162         <Item xsi:type="EVALUATION" archetype_id="openEHR-EHR-EVALUATION.problem_diagnosis-sus.v1"
163         concept_name="Problema" path="/items">
164           <Rule path="/data[at0001]/items[at0009]" max="0" />
165           <Rule path="/data[at0001]/items[at0005]" max="0" />
166           <Rule path="/data[at0001]/items[at0003]" max="0" />
167           <Rule path="/data[at0001]/items[at0004]" max="0" />
168           <Rule path="/data[at0001]/items[at0012]" max="0" />
169           <Rule path="/data[at0001]/items[at0018]" max="0" />
170           <Rule path="/data[at0001]/items[at0027]" max="0" />
171           <Rule path="/data[at0001]/items[at0030]" max="0" />
172           <Rule path="/data[at0001]/items[at0031]" max="0" />
173           <Rule path="/data[at0001]/items[at0048]" max="0" />
174           <Rule path="/data[at0001]/items[at0049]" max="0" />
175           <Rule path="/protocol[at0032]" hide_on_form="true" />
176           <Rule path="/protocol[at0032]/items[at0035]" max="0" />
177         </Item>
178       </Item>
179       <Item xsi:type="SECTION" archetype_id="openEHR-EHR-SECTION.adhoc_sus.v1"
180       concept_name="Cabealho Adhoc" max="1" path="/items" name="Exames">
181         <Item xsi:type="SECTION" archetype_id="openEHR-EHR-SECTION.adhoc_sus.v1"
182         concept_name="Cabealho Adhoc" max="1" path="/items" name="Exames solicitados">
183           <Item xsi:type="INSTRUCTION" archetype_id="openEHR-EHR-INSTRUCTION.request-lab_test.v1"
184           concept_name="Solicitao exames laboratoriais" path="/items">
```

```
185           <Rule path="/activities[at0001]" max="1" name="Solicitao" />
186           <Rule path="/activities[at0001]/description[at0009]/items[at0121.1]">
187             <constraint xsi:type="textConstraint">
188               <termQueryId terminologyID="SIGTAP" terminologyLang="Portugues" queryName="SIGTAP" />
189             </constraint>
190           </Rule>
191           <Rule path="/activities[at0001]/description[at0009]/items[at0135]" max="0" />
192           <Rule path="/activities[at0001]/description[at0009]/items[at0062]" max="0" />
193           <Rule path="/activities[at0001]/description[at0009]/items[at0064]" max="0" />
194           <Rule path="/activities[at0001]/description[at0009]/items[at0065.1]" max="0" />
195           <Rule path="/activities[at0001]/description[at0009]/items[at0068]" max="0" />
196           <Rule path="/activities[at0001]/description[at0009]/items[at0040.1]" max="0" />
197           <Rule path="/activities[at0001]/description[at0009]/items[at0144.1]" max="0" />
198           <Rule path="/activities[at0001]/description[at0009]/items[at0076]" max="0" />
199           <Rule path="/activities[at0001]/description[at0009]/items[at0078]" max="0" />
200         </Item>
201         <Item xsi:type="INSTRUCTION"
202         archetype_id="openEHR-EHR-INSTRUCTION.request-imaging_exam-sus.v1"
203         concept_name="Solicitao de exame de imagem" path="/items">
204           <Rule path="/activities[at0001]/description[at0009]" hide_on_form="true" />
205           <Rule path="/activities[at0001]/description[at0009]/items[at0121.1]"
206           name="Exame solicitado">
207             <constraint xsi:type="textConstraint">
208               <termQueryId terminologyID="SIGTAP" terminologyLang="Portugues" queryName="SIGTAP"/>
209             </constraint>
210           </Rule>
211           <Rule path="/activities[at0001]/description[at0009]/items[at0135.1]" max="0" />
212           <Rule path="/activities[at0001]/description[at0009]/items[at0062]" max="0" />
213           <Rule path="/activities[at0001]/description[at0009]/items[at0064]" max="0" />
214           <Rule path="/activities[at0001]/description[at0009]/items[at0065]" max="0" />
215           <Rule path="/activities[at0001]/description[at0009]/items[at0068.1]" max="0" />
216           <Rule path="/activities[at0001]/description[at0009]/items[at0040]" max="0" />
217           <Rule path="/activities[at0001]/description[at0009]/items[at0144]" max="0" />
218           <Rule path="/activities[at0001]/description[at0009]/items[at0076]" max="0" />
219           <Rule path="/activities[at0001]/description[at0009]/items[at0078.1]" max="0" />
220           <Rule path="/activities[at0001]/description[at0009]/items[at0.2]" max="0" />
221         </Item>
222       </Item>
223       <Item xsi:type="SECTION" archetype_id="openEHR-EHR-SECTION.adhoc_sus.v1"
224       concept_name="Cabealho Adhoc" max="1" path="/items" name="Exames avaliados">
225         <Item xsi:type="OBSERVATION" archetype_id="openEHR-EHR-OBSERVATION.lab_test.v1"
226         concept_name="Laboratory test" max="1" path="/items" name="Exames laboratoriais">
227           <Rule path="/data[at0001]" hide_on_form="true" />
228           <Rule path="/data[at0001]/events[at0002]" hide_on_form="true" />
229           <Rule path="/data[at0001]/events[at0002]/data[at0003]" hide_on_form="true" />
230           <Rule path="/data[at0001]/events[at0002]/data[at0003]/items[at0005]"name="Nome do exame">
231             <constraint xsi:type="textConstraint">
232               <termQueryId terminologyID="SIGTAP" terminologyLang="Portugues" queryName="SIGTAP"/>
233             </constraint>
234           </Rule>
235           <Rule path="/data[at0001]/events[at0002]/data[at0003]/items[at0077]" max="0" />
236           <Rule path="/data[at0001]/events[at0002]/data[at0003]/items[at0073]" max="0" />
237           <Rule path="/data[at0001]/events[at0002]/data[at0003]/items[at0078]" max="0" />
238           <Rule path="/data[at0001]/events[at0002]/data[at0003]/items[at0057]" max="0" />
239           <Rule path="/data[at0001]/events[at0002]/data[at0003]/items[at0010]" max="0" />
240         </Item>
241         <Item xsi:type="OBSERVATION" archetype_id="openEHR-EHR-OBSERVATION.imaging-sus.v1"
242         concept_name="Exame de imagem" path="/items">
243           <Rule path="/data[at0001]" hide_on_form="true" />
244           <Rule path="/data[at0001]/events[at0002]" hide_on_form="true" />
245           <Rule path="/data[at0001]/events[at0002]/data[at0003]" hide_on_form="true" />
246           <Rule path="/data[at0001]/events[at0002]/data[at0003]/items[at0020]" max="0" />
247           <Rule path="/data[at0001]/events[at0002]/data[at0003]/items[at0024]" max="0" />
248           <Rule path="/data[at0001]/events[at0002]/data[at0003]/items[at0026]" max="0" />
249           <Items xsi:type="CLUSTER" archetype_id="openEHR-EHR-CLUSTER.imaging-sus.v1"
```

```
250           concept_name="Detalhes de imagem"
251           path="/data[at0001]/events[at0002]/data[at0003]/items[at0025]">
252             <Rule path="/items[at0001]" max="0" />
253             <Rule path="/items[at0010]">
254               <constraint xsi:type="textConstraint">
255                 <termQueryId terminologyID="SIGTAP" terminologyLang="Portugues"
256                 queryName="SIGTAP" />
257               </constraint>
258             </Rule>
259             <Rule path="/items[at0011]" max="0" />
260           </Items>
261         </Item>
262       </Item>
263     </Item>
264     <Item xsi:type="ACTION" archetype_id="openEHR-EHR-ACTION.procedure.v1"
265     concept_name="Procedimento realizado" path="/items">
266       <Rule path="/description[at0001]/items[at0002]">
267         <constraint xsi:type="textConstraint">
268           <termQueryId terminologyID="SIGTAP" terminologyLang="Portugues" queryName="SIGTAP" />
269         </constraint>
270       </Rule>
271       <Rule path="/description[at0001]/items[at0014]" max="0" />
272       <Rule path="/description[at0001]/items[at0051]" max="0" />
273       <Rule path="/description[at0001]/items[at0049]" max="0" />
274       <Rule path="/description[at0001]/items[at0030]" max="0" />
275       <Rule path="/description[at0001]/items[at0048]" max="0" />
276       <Rule path="/description[at0001]/items[at0004]" max="0" />
277       <Rule path="/description[at0001]/items[at0018]" max="0" />
278       <Rule path="/description[at0001]/items[at0015]" max="0" />
279       <Rule path="/description[at0001]/items[at0006]" max="0" />
280       <Rule path="/description[at0001]/items[at0058]" max="0" />
281       <Rule path="/description[at0001]/items[at0005]" max="0" />
282       <Rule path="/description[at0001]/items[at0013]" max="0" />
283     </Item>
284     <Item xsi:type="ACTION" archetype_id="openEHR-EHR-ACTION.procedure-sus-PIC.v1"
285     concept_name="Praticas integradas e complementares" path="/items">
286       <Rule path="/description[at0001]/items[at0014]" max="0" />
287       <Rule path="/description[at0001]/items[at0059]" max="0" />
288       <Rule path="/description[at0001]/items[at0051]" max="0" />
289       <Rule path="/description[at0001]/items[at0049]" max="0" />
290       <Rule path="/description[at0001]/items[at0030]" max="0" />
291       <Rule path="/description[at0001]/items[at0048]" max="0" />
292       <Rule path="/description[at0001]/items[at0004]" max="0" />
293       <Rule path="/description[at0001]/items[at0018]" max="0" />
294       <Rule path="/description[at0001]/items[at0015]" max="0" />
295       <Rule path="/description[at0001]/items[at0006]" max="0" />
296       <Rule path="/description[at0001]/items[at0058]" max="0" />
297       <Rule path="/description[at0001]/items[at0005]" max="0" />
298       <Rule path="/description[at0001]/items[at0013]" max="0" />
299       <Rule path="/description[at0001]/items[at0.60]" max="0" />
300     </Item>
301     <Item xsi:type="ADMIN_ENTRY" archetype_id="openEHR-EHR-ADMIN_ENTRY.patient_discharge-sus.v1"
302     concept_name="Tipo de Sada " max="1" path="/items" name="Conduta/desfecho">
303       <Rule path="/data[at0001]/items[at0003]" hide_on_form="true" />
304       <Rule path="/data[at0001]/items[at0003]/items[at0010]" max="0" />
305       <Rule path="/data[at0001]/items[at0003]/items[at0006]" max="0" />
306       <Rule path="/data[at0001]/items[at0007]" max="0" />
307     </Item>
308     <Item xsi:type="INSTRUCTION" archetype_id="openEHR-EHR-INSTRUCTION.request-follow_up.v1"
309     concept_name="Solicitao de seguimento" path="/items">
310       <Rule path="/activities[at0001]" hide_on_form="true" />
311       <Rule path="/activities[at0001]/description[at0009]" hide_on_form="true" />
312       <Rule path="/activities[at0001]/description[at0009]/items[at0135.1]" max="0" />
313       <Rule path="/activities[at0001]/description[at0009]/items[at0062.1]" max="0" />
314       <Rule path="/activities[at0001]/description[at0009]/items[at0064.1]" max="0" />
```

```
315         <Rule path="/activities[at0001]/description[at0009]/items[at0065.1]" max="0" />
316         <Rule path="/activities[at0001]/description[at0009]/items[at0068.1]" max="0" />
317         <Rule path="/activities[at0001]/description[at0009]/items[at0040.1]" max="0" />
318         <Rule path="/activities[at0001]/description[at0009]/items[at0144.1]" max="0" />
319         <Rule path="/activities[at0001]/description[at0009]/items[at0076.1]" max="0" />
320         <Rule path="/activities[at0001]/description[at0009]/items[at0078.1]" max="0" />
321       </Item>
322       <Item xsi:type="INSTRUCTION" archetype_id="openEHR-EHR-INSTRUCTION.request-referral.v1"
323       concept_name="Solicitao de encaminhamento" path="/items">
324         <Rule path="/activities[at0001]/description[at0009]/items[at0135.1]" max="0" />
325         <Rule path="/activities[at0001]/description[at0009]/items[at0062.1]" max="0" />
326         <Rule path="/activities[at0001]/description[at0009]/items[at0064.1]" max="0" />
327         <Rule path="/activities[at0001]/description[at0009]/items[at0065.1]" max="0" />
328         <Rule path="/activities[at0001]/description[at0009]/items[at0068.1]" max="0" />
329         <Rule path="/activities[at0001]/description[at0009]/items[at0040.1]" max="0" />
330         <Rule path="/activities[at0001]/description[at0009]/items[at0144.1]" max="0" />
331         <Rule path="/activities[at0001]/description[at0009]/items[at0076.1]" max="0" />
332         <Rule path="/activities[at0001]/description[at0009]/items[at0078.1]" max="0" />
333       </Item>
334       <Item xsi:type="SECTION" archetype_id="openEHR-EHR-SECTION.adhoc_sus.v1"
335       concept_name="Cabealho Adhoc" max="1" path="/items" name="Anexos" />
336     </Content>
337     <Context />
338   </definition>
339   <integrity_checks xsi:type="ArchetypeIntegrity" archetype_id="openEHR-EHR-COMPOSITION.encounter.v1">
340     <digest id="MD5-CAM-1.0.1">649B46CE7C508CF5108A0A955BD22CFF</digest>
341   </integrity_checks>
342   <integrity_checks xsi:type="ArchetypeIntegrity" archetype_id="openEHR-EHR-ADMIN_ENTRY.admission-sus.v1
        ">
343     <digest id="MD5-CAM-1.0.1">6D0AD4EE9EA9634AF390E96D09F12FA8</digest>
344   </integrity_checks>
345   <integrity_checks xsi:type="ArchetypeIntegrity" archetype_id="openEHR-EHR-SECTION.adhoc_sus.v1">
346     <digest id="MD5-CAM-1.0.1">4A9A75A1D94FB883305EF4777C116908</digest>
347   </integrity_checks>
348   <integrity_checks xsi:type="ArchetypeIntegrity" archetype_id="openEHR-EHR-OBSERVATION.body_weight.v1"
        >
349     <digest id="MD5-CAM-1.0.1">1934DEB9EA72983B55F5461D37777A0E</digest>
350   </integrity_checks>
351   <integrity_checks xsi:type="ArchetypeIntegrity" archetype_id="openEHR-EHR-OBSERVATION.height.v1">
352     <digest id="MD5-CAM-1.0.1">3E91B8BE4970E92DBAF71D22E1078973</digest>
353   </integrity_checks>
354   <integrity_checks xsi:type="ArchetypeIntegrity" archetype_id="openEHR-EHR-EVALUATION.
        vaccination_summary-sus.v1">
355     <digest id="MD5-CAM-1.0.1">5B9EEEC921DEC2D5583CCA1D1BA401EA</digest>
356   </integrity_checks>
357   <integrity_checks xsi:type="ArchetypeIntegrity" archetype_id="openEHR-EHR-EVALUATION.
        aleitamento_materno.v1">
358     <digest id="MD5-CAM-1.0.1">58D5C093982EFB4C1464B189EBBBEB5D</digest>
359   </integrity_checks>
360   <integrity_checks xsi:type="ArchetypeIntegrity" archetype_id="openEHR-EHR-EVALUATION.pregnancy-sus.v1"
        >
361     <digest id="MD5-CAM-1.0.1">26DA8A36A24A6C26B289A6DFBF0A49CB</digest>
362   </integrity_checks>
363   <integrity_checks xsi:type="ArchetypeIntegrity" archetype_id="openEHR-EHR-EVALUATION.
        modalidade_atencao_domiciliar.v1">
364     <digest id="MD5-CAM-1.0.1">980EBC70817064E3D8D124057661A1C9</digest>
365   </integrity_checks>
366   <integrity_checks xsi:type="ArchetypeIntegrity" archetype_id="openEHR-EHR-EVALUATION.problem_diagnosis
        -sus.v1">
367     <digest id="MD5-CAM-1.0.1">F8E3A71C3C0CFB4D1458B6FEFEB725C2</digest>
368   </integrity_checks>
369   <integrity_checks xsi:type="ArchetypeIntegrity" archetype_id="openEHR-EHR-INSTRUCTION.request-lab_test
        .v1">
370     <digest id="MD5-CAM-1.0.1">0CD9CA4F15B69EBA42699DDE034A8BB0</digest>
371   </integrity_checks>
```

```
<integrity_checks xsi:type="ArchetypeIntegrity" archetype_id="openEHR-EHR-INSTRUCTION.request-imaging_exam-sus.v1">
  <digest id="MD5-CAM-1.0.1">5BF47100BFBFA8F8988CE337BD5EBBF5</digest>
</integrity_checks>
<integrity_checks xsi:type="ArchetypeIntegrity" archetype_id="openEHR-EHR-OBSERVATION.lab_test.v1">
  <digest id="MD5-CAM-1.0.1">AA72B9E5B66B27731CB2E3FE3F42D9F9</digest>
</integrity_checks>
<integrity_checks xsi:type="ArchetypeIntegrity" archetype_id="openEHR-EHR-OBSERVATION.imaging-sus.v1">
  <digest id="MD5-CAM-1.0.1">5B79D93AAFF8167F0D55369817F28254</digest>
</integrity_checks>
<integrity_checks xsi:type="ArchetypeIntegrity" archetype_id="openEHR-EHR-CLUSTER.imaging-sus.v1">
  <digest id="MD5-CAM-1.0.1">D402238C16A27BAC0D4937BA404CB4C9</digest>
</integrity_checks>
<integrity_checks xsi:type="ArchetypeIntegrity" archetype_id="openEHR-EHR-ACTION.procedure.v1">
  <digest id="MD5-CAM-1.0.1">457962974FA0AB42EDD08064B980BD25</digest>
</integrity_checks>
<integrity_checks xsi:type="ArchetypeIntegrity" archetype_id="openEHR-EHR-ACTION.procedure-sus-PIC.v1">
  <digest id="MD5-CAM-1.0.1">72F73FB9F9EC950157A97D5B1A31ABD9</digest>
</integrity_checks>
<integrity_checks xsi:type="ArchetypeIntegrity" archetype_id="openEHR-EHR-ADMIN_ENTRY.patient_discharge-sus.v1">
  <digest id="MD5-CAM-1.0.1">E79C23CFC9049AD62448CBCF347F0109</digest>
</integrity_checks>
<integrity_checks xsi:type="ArchetypeIntegrity" archetype_id="openEHR-EHR-INSTRUCTION.request-follow_up.v1">
  <digest id="MD5-CAM-1.0.1">53DD571B8766A978B6A53B65F1330D1F</digest>
</integrity_checks>
<integrity_checks xsi:type="ArchetypeIntegrity" archetype_id="openEHR-EHR-INSTRUCTION.request-referral.v1">
  <digest id="MD5-CAM-1.0.1">793934D4F1BE47DE122F21325F2FDC64</digest>
</integrity_checks>
</template>
```